

X . s y s t e m s . p r e s s

X.systems.press ist eine praxisorientierte
Reihe zur Entwicklung und Administration von
Betriebssystemen, Netzwerken und Datenbanken.

Klaus Braune · Joachim Lammarsch
Marion Lammarsch

L^AT_EX

Basissystem, Layout, Formelsatz

Mit 35 Abbildungen und 48 Tabellen

 Springer

Klaus Braune
Rechenzentrum
Universität Karlsruhe
Postfach 69 80
76128 Karlsruhe
klaus.braune@rz.uni-karlsruhe.de

Joachim Lammarsch
Universitätsrechenzentrum
Universität Heidelberg
Im Neuenheimer Feld 293
69120 Heidelberg
joachim.lammarsch@urz.uni-heidelberg.de

Marion Lammarsch
Psychologisches Institut
Universität Heidelberg
Hauptstr. 47–51
69117 Heidelberg
marion.lammarsch@psychologie.uni-heidelberg.de

Bibliografische Information der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen
Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über
<http://dnb.ddb.de> abrufbar.

ISSN 1611-8618
ISBN-10 3-540-00718-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-00718-0 Springer Berlin Heidelberg New York

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

Springer ist ein Unternehmen von Springer Science+Business Media
springer.de

© Springer-Verlag Berlin Heidelberg 2006
Printed in Germany

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften. Text und Abbildungen wurden mit größter Sorgfalt erarbeitet. Verlag und Autor können jedoch für eventuell verbliebene fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.

Satz: Druckfertige Daten der Autoren
Herstellung: LE-TEX, Jelonek, Schmidt & Vöckler GbR, Leipzig
Umschlaggestaltung: KünkelLopka Werbeagentur, Heidelberg
Gedruckt auf säurefreiem Papier 33/3142 YL – 5 4 3 2 1 0

Vorwort

Perfekter Satz, perfekter Satz mathematischer Formeln, unzählige Schriften, in den unterschiedlichsten Sprachen, Satz auch von rechts nach links und zeichenweise von oben nach unten, Musiksatz, und dazu noch kostenlos für alle gängigen Betriebssysteme verfügbar – die Laudatio findet kein Ende.

Es handelt sich hierbei nicht um eine gut gemachte Werbung (von weltweit verkaufter Textverarbeitungssoftware), sondern wir sprechen von einer über ein vierteljahrhundert alten Software – von $\text{\TeX}$ und dem dazu gehörigen Makropaket $\text{\LaTeX}$. Dieses $\text{\LaTeX}$, das mittlerweile auf Grund seiner Mächtigkeit so beliebt ist, dass das Programm $\text{\TeX}$, welches im Hintergrund den eigentlichen Satz macht, fast in Vergessenheit geraten ist. Man spricht nur noch von $\text{\LaTeX}$.

Es gibt weitere Produkte im Umfeld von $\text{\LaTeX}$ und $\text{\TeX}$, die erfolgreich von Anwendern eingesetzt werden. Sei es in Form von Bildschirm- oder Druckertreibern, bei der Erzeugung von Grafiken, beim Erstellen von HTML- oder PDF-Seiten, bei dem Entwurf von Schriften und vielem anderem mehr – immer wieder gibt es in der $\text{\TeX}$ -Welt interessante neue Software, die verwendet werden kann. Selbst eine in Java geschriebene $\text{\TeX}$ -Version ist mittlerweile vorhanden.

Das Interesse der $\text{\LaTeX}$ -Anwender hat sich gewandelt. Stand früher die Typografie im Vordergrund, so wollen sie sich heute nicht umfangreich in ein neues System einarbeiten, sondern möglichst sofort Texte erstellen. Der Aufwand darf für sie nicht größer sein als bei anderen Textsystemen. Für diese Ansprüche gibt es mittlerweile freie, luxuriöse Oberflächen, beispielsweise KILE unter Unix (KDE) und andere. Unter Windows kennt man (allerdings als Shareware) WINEDT, von dessen Funktionalität jedoch das freie $\text{\TeX}$ NICCENTER nicht mehr viel vermissen lässt. Diese

Liste könnte, auch für nicht explizit aufgelistete Betriebssysteme, fortgesetzt werden. Selbst wenn $\text{\LaTeX}$ keine sogenannte WYSIWYG¹-Software ist, mit den richtigen Hilfsmitteln ist ein WYSIWYG-ähnliches Arbeiten möglich.

Gleichzeitig stellt man jedoch immer wieder fest, dass die meisten Anwender nur Teilaspekte des riesigen Gebiets überblicken (und auch die eher rudimentär) und sich der enormen Möglichkeiten, die $\text{\LaTeX}$, $\text{\TeX}$ und Co. bieten, nicht bewusst sind. Das trifft natürlich noch viel mehr auf die (noch) Nicht-Anwender zu, die sich lieber mit einem Standard-Textformatierer abmühen, statt auf bessere Software zurückzugreifen.

Wenn ein Verlag wegen eines neuen $\text{\LaTeX}$ -Buchs nachfragt, dann ist der erste Gedanke:

Warum? Wofür?

Es gibt doch schon genügend Bücher zu diesem Thema.

Betrachtet man die Literaturliste am Ende dieses Buchs, dann kann man diesen Gedanken sehr gut nachvollziehen. Wenn allerdings der Lektor dann seine Idee vorstellt, nicht nur ein $\text{\LaTeX}$ -Buch zu schreiben, sondern ein umfassendes Gesamtwerk zu $\text{\LaTeX}$ zu schaffen, drei Bände, die die Grundlagen, die Makropakete, die begleitende Software abdecken sollen, dann ist das eine Herausforderung, über die sich ein Nachdenken lohnt. In den Sinn kommt das vierbändige Werk zu $\text{\TeX}$ von Stefan Bechtolsheim, das beim Springer-Verlag in den USA entstanden ist. Und wie man hier sieht: Wir konnten der Herausforderung nicht widerstehen.

$\text{\LaTeX}$, Makropakete und Software

$\text{\LaTeX}$ – genauer $\text{\LaTeX} 2_{\epsilon}$ – zu beschreiben ist schon nicht einfach. Es stellt sich immer die Frage, ob eine Spezialität noch erwähnenswert ist, oder ob man zu sehr ins Detail geht. Dann folgte ein Jahr, in welchem wir versuchten, einen Überblick über die vorhandenen Makropakete zu bekommen. Wir testeten die unterschiedlichsten Pakete, fanden auch einige Fehler, die wir den Autoren mitteilten. Aber als wir unsere Liste abgearbeitet hatten, stellten wir fest, dass mittlerweile so viele neue Pakete und neue Versionen veröffentlicht worden waren, dass wir von Neuem hätten anfangen können. Bei der zusätzlich vorhandenen Software stellte sich die Situation ähnlich dar. Alles vorzustellen und zu beschreiben, hätte

¹ What you see is what you get – oder ein wenig sarkastisch formuliert, man bekommt nicht mehr als man sieht.

mehr als drei Bände gefüllt – die doppelte Anzahl hätte wahrscheinlich nicht ausgereicht, von der benötigten Zeit ganz zu schweigen. Im nächsten Ansatz versuchten wir zu eruieren, was in unserem Umfeld benötigt wird. Auch dieser Versuch der Eingrenzung misslang kläglich. Schon in unserem universitären Bereich ist der Einsatz so vielfältig, dass die Auswahl sehr schwer fiel. Und es erinnert wiederum an Stefan Bechtolsheim, der mit seinem Buchprojekt jahrelang nicht zu einem Ende kam – es ging uns nicht anders. Wir konnten uns nicht entscheiden, was tun?

Daher entschlossen wir uns, das zu behandeln, was wir für notwendig hielten, was oft nachgefragt wird – und ganz wichtig, was uns besonders gut gefiel. Man sieht, die Entscheidung wurde sehr demokratisch getroffen, trotzdem hoffen wir, dass für jeden etwas Nützliches und Interessantes zu finden ist. Wir haben uns für folgende Aufteilung entschieden:

Band 1, $\LaTeX$ – Das Basissystem: Dieser Band beschreibt, was in einer Standard- $\LaTeX$ -Verteilung enthalten ist. Darüber hinaus, weil fast immer benötigt, die Anpassung an europäische Sprachen, insbesondere die deutsche Sprache. Die PostScript-Zeichensätze und Informationen zu den Oberflächen verschiedener Betriebssysteme sind ebenfalls enthalten.

Band 2, $\LaTeX$ – Fonts, Layout, Markup: Interessante $\LaTeX$ -Erweiterungen werden beschrieben, welche noch nicht zum Standard gehören. Pakete mit eignen Dokumentklassen und ihre typografischen Aspekte werden beleuchtet. Ausführliche Beispiele und Tipps für Diplom-, Doktor- und Habilitationsarbeiten sowie Modifikationen der Layouts.

Band 3, $\LaTeX$ – Werkzeuge, Grafik, WWW: In diesem Band findet alles Platz, was weit über die normalen $\LaTeX$ -Anwendungen hinausgeht. Auch finden sich hier die Software-Pakete, welche im eigentlichen Sinne nicht direkt mit $\LaTeX$ zu tun haben, aber doch für manchen Anwender von Interesse sind. Ferner soll auch das Erwähnung finden, was in den ersten beiden Bänden vergessen wurde.

Man muss diesen ersten Band nicht von vorne bis hinten lesen, um das erste $\LaTeX$ -Dokument zu erstellen. Anfängern und auch Anwendern mit ersten Grundkenntnissen empfehlen wir den Teil I. Die nachfolgenden Kapitel können je nach Bedarf zu Rate gezogen werden.

Typografische Konventionen

Damit Sie die Verwendung der Informationen in diesem Buch klar zuordnen können, werden die folgenden Konventionen eingehalten:

- $\LaTeX$ -Befehle, -Umgebungen und Optionen werden in einer äquidistanten Schrift dargestellt.² Beispiele wären hier der Befehl `\chapter`, die Umgebung `tabular` bzw. der Aufruf der Umgebung `\begin{tabular}`.
- $\LaTeX$ -Pakete oder Klassen-Dateien werden in einer serifenlosen Schrift ausgezeichnet, beispielsweise `article` oder `german`.
- $\LaTeX$ -Befehlsstrukturen sind grau hinterlegt. Argumente, die zu ersetzen sind, werden kursiv angegeben und in spitze Klammern eingeschlossen.³ Optionale Argumente werden zusätzlich in `[` und `]` eingeschlossen.

```
\begin{eqnarray}
  [<Linke Teilformel>]
    [& [<Mittlere Teilformel>]
      [& [<Rechte Teilformel>] \\
    ... ]]]]
\end{eqnarray}
```

- Textteile, die $\LaTeX$ -Eingaben enthalten, werden in einer etwas kleineren äquidistanten Schrift und mit einer Zeilennummer versehen dargestellt.

```
8 Hier kommt eine Eingabe:
9 Ein kurzer Beispieltext.
```

- Bei Programmbeispielen werden Eingabe und Ausgabe gegenübergestellt. Sie können nebeneinander dargestellt oder bei Bedarf untereinander gesetzt werden. Die $\LaTeX$ -Eingabe wird auch hier in einer etwas kleineren äquidistanten Schrift und mit Zeilennummern versehen dargestellt. Die Ausgabe erhält eine rechteckige Box.

```
7 \begin{document}
8 Ein kurzer Beispieltext.
9 \end{document}
```

Ein kurzer Beispieltext.

[Buch1/Vorwort/bspexq.tex]

Die untereinander gestellte Variante hat folgende Form:

```
8 Ein langer Beispieltext. Ein langer Beispieltext. Ein
9 langer Text Text. Ein langer Beispieltext. Ein
10 langer Beispieltext. Ein langer Text Text.
```

² Teilweise müssen sie wegen der begrenzten Buchbreite beim Zeilenumbruch getrennt werden. Diese Trennung ist natürlich nicht Bestandteil des Befehls.

³ Die Schreibweise ist bei Lesen zunächst abschreckend, wenn man aber schnell etwas nachschlagen möchte, ist diese Art der Markierung gut erkennbar.

Ein langer Beispieltext. Ein langer Beispieltext. Ein langer Text Text.
 Ein langer Beispieltext. Ein langer Beispieltext. Ein langer Text Text.

[Buch1/Vorwort/bspex.tex]

- Um das Problem unterschiedlicher Eingabe-Zeichensätze zu vermeiden, werden in allen Beispielen Umlaute und Eszett folgendermaßen kodiert: "A "a "O "o "U "u "s entsprechen jeweils Ä ä Ö ö Ü ü ß. Diese Kodierung können Sie mit dem Makropaket ngerman bzw. mit dem Aufruf des babel-Pakets

```
\usepackage[ngerman]{babel}
```

ebenfalls verwenden.

Anmerkungen zu den Beispielen

Möchte man eine Menge von Makropaketen beschreiben, die die Möglichkeiten von $\LaTeX$ erweitern, müssten diese eigentlich alle gleichzeitig geladen werden. Das ist nicht die beste Lösung, da dies die Möglichkeiten des $\TeX$ -Systems an einigen Stellen überfordert. Beispielsweise ist bei $\TeX$ die Anzahl der Zähler auf 256 begrenzt. Manche Pakete legen bis zu 20 eigene Zähler an, da ist schnell das Ende erreicht. Einige Pakete erfordern zwingend den Einsatz von PostScript, also den dvips-Druckertreiber. Andere arbeiten besser mit PDF-Dateien.

Zusammenfassend kann man feststellen, dass es sehr große Probleme bereitet, einen vollständigen Band einer $\LaTeX$ -Beschreibung auf einmal zu setzen. Wir haben uns daher für eine andere Vorgehensweise entschieden. Alle Beispiele sind vollständige $\LaTeX$ -Dokumente und werden aus dem Buch heraus in separate Dateien geschrieben. Diese Eingabedateien werden in getrennten $\LaTeX$ -Läufen verarbeitet und die Ausgabe nach PDF konvertiert. Damit kann diese Ausgabe wiederum als Bild, d. h. mit dem Paket graphicx in das Buch eingebunden werden. Dies hat den Vorteil, dass es nicht notwendig ist, mehr als die für das jeweilige Beispiel benötigten Pakete zu laden. Außerdem stehen damit alle Beispiele als eigenständige Eingabedateien zur Verfügung. Um dem Leser das Abtippen zu ersparen, sind sie unter der Adresse

<http://lammarsch.uni-hd.de/lbssv>

zum Herunterladen verfügbar.

Über die Autoren

Wenn wir auch heute nicht mehr zu den Aktiven in der $\text{T}_{\text{E}}\text{X}$ -Welt zählen, so haben wir uns seit vielen Jahren mit $\text{T}_{\text{E}}\text{X}$ und $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ beschäftigt. $\text{T}_{\text{E}}\text{X}$ ist ganz bewusst an erster Stelle aufgeführt, denn am Anfang gab es noch kein $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$. Damit wird offensichtlich, es handelt sich um eine sehr lange Zeit, zusammengezählt kommen wir auf über 60 Jahre. Wir unterrichteten (und tun es immer noch) $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ und früher auch $\text{T}_{\text{E}}\text{X}$. Wir beraten bei $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Fragen in den Universitäten, an denen wir beschäftigt sind. Eigene Unterlagen und private Korrespondenz werden von uns mit $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ verfasst. Wir waren (ohne Anspruch auf Vollständigkeit und immer nur einfach aufgeführt) in folgenden Funktionen tätig: Koordinator der deutschsprachigen $\text{T}_{\text{E}}\text{X}$ -Interessenten, DANTE-Gründungsmitglied, DANTE-Präsident, DANTE-Beirat, TUG-site coordinator, TUG spec. vice-president, Tutor, Autor und mit der Durchführung der Europäischen $\text{T}_{\text{E}}\text{X}$ -Tagung 1999 aus dem aktiven Geschehen zurückgetreten.

Trotzdem war es ein Vergnügen, vom Lektor Dr. Frank Schmidt vom Springer Verlag Heidelberg reaktiviert zu werden. Es ist schön, noch einmal etwas für die Verbreitung dieser wundervollen Software tun zu können, wie wir es in der Vergangenheit Jahrzehnte lang getan haben. Auch die Mitarbeit bei DANTE, Deutschsprachige Anwendervereinigung e. V. der lokalen Benutzergruppe, und bei der TUG, $\text{T}_{\text{E}}\text{X}$ Users Group, ist nicht nur Arbeit, sondern dient der Unterstützung einer Software, die weltweit ihresgleichen sucht. Wir werden einen Teil der Erträge aus diesem Buchprojekt als Spende an DANTE e. V. weiterleiten, damit weitere Entwicklungen in diesem Bereich gefördert werden können.

Unser Dank

Unser großer Dank geht an ...

Dr. Frank Schmidt vom Springer Verlag Heidelberg für die Idee zu diesem Buchprojekt, für die angenehme Betreuung, die anregenden Gespräche, die uns immer wieder motiviert haben. Leider hat er zu unserem großen Bedauern vor einem Jahr den Verlag verlassen.

Frau Jutta Maria Fleschutz und Herrn Hermann Engesser vom Springer Verlag, die danach das Projekt übernommen haben, für ihre Geduld – auch wenn sie uns gegen Ende die „Folterwerkzeuge“ gezeigt haben, damit der erste Band endlich fertig wurde.

All diejenigen, die uns bei Problemen und Korrekturen hilfreich zur Seite standen. Hier sind insbesondere Dieter Berger und Günter Partosch zu

nennen. Für alle ganz bestimmt noch vorhandenen Fehler sind selbstverständlich die Autoren allein verantwortlich.

Dem Vorstand von DANTE e. V., stellvertretend für alle, die ehrenamtlich und in ihrer Freizeit für die Entwicklung und Verbreitung der Software sorgen, die uns allen so am Herzen liegt.

Leslie Lamport, Frank Mittelbach, Chris Rowley und Rainer Schöpf stellvertretend für das ganze L^AT_EX 3 Team. Dem ersten für die Idee, das Makropaket zu entwickeln; den anderen war es gelungen, Lamport zu überzeugen, dass man L^AT_EX 2.09 verbessern muss. Danach haben sie sich jahrelang um die Entwicklung von L^AT_EX 2_ε verdient gemacht.

Der allergrößte Dank geht jedoch an Prof. Donald E. Knuth, der uns allen eine wahrhaft überragende Software unentgeltlich und uneingeschränkt überlassen hat. Der es geschafft hat, damit Menschen zusammenzubringen (und das heute nach über 25 Jahren immer noch), um für eine Idee Energie und Zeit zu investieren und sich in einer Gemeinschaft (der *T_EX-community*) für perfekten Schriftsatz und Typografie einzusetzen. Der trotz aller weltweiten Ehrungen und Auszeichnungen ein ganz bescheidener Mensch geblieben ist, mit dem man am Küchentisch in seinem Haus in Stanford die T_EX-Probleme diskutieren kann.

Bitte um Rückmeldung

Wir haben uns bemüht, alle Pakete in den neuesten Versionen vollständig zu beschreiben. Dennoch ist es möglich, dass wir Optionen übersehen oder falsch verstanden haben. Kleinere Fehler, Unvollständigkeiten und Unklarheiten sind bei einem so umfangreichen Werk in der ersten Auflage nicht zu vermeiden. Um deren Behebung sind wir natürlich bemüht. Wir möchten Sie deshalb bitten, uns solche Mängel mitzuteilen, damit wir sie in der nächsten Version verbessern können – Herzlichen Dank.

Klaus Braune
Joachim Lammarsch
Marion Lammarsch

Karlsruhe
Heidelberg
Heidelberg
im März 2006

Inhaltsverzeichnis

Teil I Einstieg

1	Einleitung	3
1.1	T _E X	4
1.2	L ^A T _E X	7
1.3	Zusätzliche Software	9
1.4	Anwendervereinigungen und Software-Bezugsquellen	10
1.4.1	T _E X Users Group	10
1.4.2	DANTE e. V.	11
1.4.3	Software-Verteilung	11
2	Erste Schritte	13
2.1	Ein erstes Beispiel	13
2.2	Bearbeitung eines Dokuments	14
2.2.1	Befehle	19
2.2.2	Umgebungen	23
2.2.3	Gruppen	26
2.2.4	Kommentare	27
2.3	Struktur eines Dokuments	28
2.3.1	Texteingabe	28
2.3.2	Dokumentklassen	33
2.3.3	Seitenstil	34
2.3.4	Zeilen- und Seitenumbruch	35
2.4	Gliederung von Texten	37
2.4.1	Kapitel und Abschnitte	37
2.4.2	Inhaltsverzeichnis	38
2.4.3	Aufzählungen	39
2.4.4	Tabellen	41
2.5	Formelsatz	44
2.5.1	Formeln im Text	45

2.5.2	Abgesetzte Formeln	45
2.6	Fazit	46
3	Dokumentstrukturen	47
3.1	Dokumentlayout.....	47
3.1.1	Dokumentoptionen	49
3.2	Seitenlayout	51
3.2.1	Deklarationen von Kopf- und Fußzeilen	52
3.2.2	Seitennummerierung	53
3.3	Makropakete	54
3.4	Zusammenfassungen	55
3.5	Titelseite	56
3.5.1	titlepage-Umgebung	56
3.5.2	maketitle-Befehl	57
3.6	Gliederungsbefehle	59
3.6.1	Ändern der Nummerierung	61
3.6.2	Ändern von Überschrift-Ebene und Schriftart....	63
3.6.3	Spezielle Befehle für die Klasse book	67
3.7	Anhang	68
3.8	Inhaltsverzeichnis	68
4	Spezielle Textstrukturen	71
4.1	Aufzählungen	71
4.1.1	enumerate-Umgebung	71
4.1.2	itemize-Umgebung.....	72
4.1.3	Allgemeine Listen	74
4.2	Tabellen	77
4.2.1	tabbing-Umgebung.....	78
4.2.2	tabular-Umgebung.....	82
4.3	Textverschiebungen.....	94
4.3.1	Zentrierter Text	95
4.3.2	Einseitig rechts- oder linksbündiger Text	95
4.3.3	Zitate	96
4.3.4	Gedichte.....	97
4.4	Fußnoten und Randnotizen	98
4.4.1	Fußnoten	98
4.4.2	Randnotizen	101
4.5	Gleitobjekte	102
4.5.1	Layout-Parameter für Gleitobjekte.....	104
4.5.2	Algorithmus zur Ausgabe von Gleitobjekten	106
4.5.3	Beschriftung von Gleitobjekten	106
4.6	Querverweise	108
5	Eigene Textstrukturen	111
5.1	Schriften.....	111

5.1.1	$\text{\TeX}$ -Zeichensatz	111
5.1.2	$\text{\LaTeX}$ -Zeichensatz	112
5.1.3	Hervorhebung von Text	119
5.2	Akzente und Sonderzeichen	120
5.2.1	Akzente	120
5.2.2	Weitere Sonderzeichen	121
5.2.3	Angloamerikanische Anführungszeichen	121
5.2.4	Bindestriche, Gedankenstriche und Andere	122
5.2.5	Erweiterung zur Trennung	123
5.3	Boxen	124
5.3.1	Horizontale Boxen	124
5.3.2	Absatzboxen	129
5.3.3	Rechteckboxen	130
5.4	Maßeinheiten, Abstände und Zähler	130
5.4.1	Maßangaben	130
5.4.2	Abstände	131
5.4.3	Längenangaben	133
5.4.4	Zähler	134
5.4.5	Ausgabe von Zählern	135
6	Mathematische Formeln	137
6.1	Umgebungen für Formeln	138
6.1.1	Formeln im Fließtext	139
6.1.2	Einzeilige abgesetzte Formeln	140
6.1.3	Mehrzeilige abgesetzte Formeln	141
6.1.4	Layout-Einstellungen für abgesetzte Formeln	144
6.2	Speziell formatierte Teile von Formeln	145
6.2.1	Hoch- und Tiefstellen von Teilformeln	145
6.2.2	Über- und Unterklammern von Teilformeln	146
6.2.3	Wurzeln	150
6.2.4	Brüche	151
6.2.5	Zeilen und Spalten	152
6.3	Symbole	155
6.3.1	Griechische Buchstaben	155
6.3.2	Kalligrafische Großbuchstaben	155
6.3.3	Schriften in Formeln	156
6.3.4	Binäre Operatoren	157
6.3.5	Relationen	158
6.3.6	Große Operatoren	161
6.3.7	Funktionen	162
6.3.8	Klammern	162
6.3.9	Verschiedene Arten von Auslassungspunkten	165
6.3.10	Allgemeine Symbole	166
6.3.11	Akzente im mathematischen Modus	167

7	Basispakete	169
7.1	Paket alltt	170
7.2	Paket doc	172
7.3	Paket exscale	173
7.4	Pakete fixltx2e und fix-cm	173
7.4.1	fixltx2e	174
7.4.2	fix-cm	175
7.5	Paket flatter	176
7.6	Paket fontenc	177
7.7	Paket graphpap	178
7.8	Paket ifthen	180
7.9	Paket inputenc	186
7.10	Paket latexsym	189
7.11	Pakete makeidx und showidx	190
7.11.1	makeidx	190
7.11.2	showidx	191
7.12	Pakete newlfont und oldlfont	193
7.12.1	oldlfont	193
7.12.2	newlfont	194
7.13	Paket shortvrb	194
7.14	Paket syntonly	196
7.15	Paket textcomp	196
7.16	Paket tracefnt	200

Teil II Erweiterungen

8	Sprachunterstützung mit babel	205
8.1	Texte in deutscher Sprache	210
8.1.1	Gemeinsame Abkürzungen und Befehle	211
8.1.2	Zusätze bei der alten Rechtschreibung	213
8.1.3	Kleines Beispiel	214
8.2	Englische Texte	216
8.3	Französische Texte	217
8.3.1	Vollständiges Beispiel	226
9	Hilfreiche Ergänzungen	229
9.1	Paket afterpage	229
9.2	Paket array	230
9.3	Paket bm	240
9.4	Paket calc	246
9.5	Paket dcolumn	248
9.6	Paket delarray	251
9.7	Paket enumerate	253
9.8	Paket fontsmpl	255

9.9	Paket ftnright	256
9.10	Paket hhline	258
9.11	Paket indentfirst	263
9.12	Paket layout	265
9.13	Paket longtable	266
9.14	Paket multicol	273
9.15	Paket rawfonts	286
9.16	Paket showkeys	286
9.17	Paket somedefs	288
9.18	Paket tabularx	290
9.19	Paket theorem	294
9.20	Paket trace	299
9.21	Paket varioref	301
9.22	Paket verbatim	309
9.23	Paket xr	313
9.24	Paket xspace	314
10	Grafik, Bilder und Farben	317
10.1	Gerätetreiber	318
10.2	Paket color	320
10.3	Pakete graphics und graphicx	328
10.3.1	Drehen von Ausgabeelementen	329
10.3.2	Skalieren von Ausgabeelementen	331
10.3.3	Einbinden von Bildern	334
10.3.4	Vereinbarungen beim Einbinden von Bildern	342
10.4	Weitere Pakete	344
10.4.1	Paket lscape	345
10.4.2	Paket epsfig	345
10.4.3	Paket pstcol	346
10.4.4	Paket keyval	346
10.4.5	Paket trig	346
11	PostScript-Zeichensätze	347
11.1	Paket avant	350
11.2	Paket bookman	351
11.3	Paket chancery	351
11.4	Paket charter	352
11.5	Paket courier	352
11.6	Paket helvet	353
11.7	Paket mathpazo	354
11.8	Paket mathptmx	356
11.9	Paket newcent	358
11.10	Paket pifont	358
11.10.1	„Zapf Dingbats“-Font	359
11.10.2	PostScript-„Symbol“-Font	362

11.11	Veraltete Pakete	365
11.11.1	Paket mathppl	365
11.11.2	Paket mathptm	365
11.11.3	Paket palatino	366
11.11.4	Paket times	366
11.11.5	Paket utopia	366
12	Erweiterter Mathematiksatz – Die Pakete der $\mathcal{A}\mathcal{M}\mathcal{S}$	367
12.1	Paket amsmath	369
12.1.1	Übersicht über Formelumgebungen	371
12.1.2	Formelnummerierung	373
12.1.3	Umgebungen für abgesetzte Formeln	380
12.1.4	Umgebungen für Teilformeln	389
12.1.5	Zusätzliche Befehle im mathematischen Modus ..	401
12.2	Weitere $\mathcal{A}\mathcal{M}\mathcal{S}$ -Pakete	420
12.2.1	Kommutative Diagramme mit amscd	420
12.2.2	Paket ambsy	421
12.2.3	Paket amsopn	423
12.2.4	Texte in Formeln mit amstext	425
12.2.5	Ergänzungen mit amxtra	426
13	Layout-Erweiterungen der $\mathcal{A}\mathcal{M}\mathcal{S}$	427
13.1	Dokumentklassen und stilistische Ergänzungen	430
13.1.1	Die Dokumentklassen der $\mathcal{A}\mathcal{M}\mathcal{S}$	430
13.1.2	Paket amsthm	433
13.1.3	Paket textcmds	439
13.1.4	Mehrere Register mit amsmid	440
13.2	Literaturverzeichnisse mit amsrefs	442
13.2.1	Namen	446
13.2.2	Literaturverzeichnisse mit BIB $\mathrm{T}_{\mathrm{E}}\mathrm{X}$	448
13.2.3	Befehle zum Aufbau von Literaturverzeichnissen	449
13.2.4	Eintragstypen	458
13.2.5	Feldattribute	469
13.2.6	Beispiel einer <code>ltb</code> -Datei	470
13.2.7	Zitieren	472
13.2.8	Anpassung von Bibliografestilen	476
13.3	Die $\mathcal{A}\mathcal{M}\mathcal{S}$ -Fonts	483
13.3.1	Paket amsfonts	484
13.3.2	Paket amssymb	485
13.3.3	„Euler Fraktur“-Font	490
13.3.4	„Euler Script“-Font	492
13.3.5	„Euler“-Font	493
13.3.6	Paket cmmib57	494

Teil III Zusätze

14 Benutzeroberflächen	499
14.1 Textorientierte Oberflächen	500
14.1.1 GNU Emacs	501
14.1.2 Emacs mit AUCTeX	509
14.2 Grafische Entwicklungsumgebungen	514
14.2.1 TeXmaker	519
14.2.2 TeXnicCenter	520
14.2.3 Kile	521
14.2.4 Fazit	522
14.3 WYSIWYG-ähnliche Oberflächen	522
14.3.1 LyX	523
14.3.2 TeXmacs	533
15 Feinheiten der Typografie	545
15.1 Einleitung	545
15.2 Abkürzungen, Akronyme, Einheiten, Himmelsrichtungen	547
15.2.1 Schreibweise von Abkürzungen	548
15.2.2 Liste gebräuchlicher Abkürzungen	549
15.3 Anführungszeichen, Zitate	550
15.3.1 Schreibweisen von Anführungszeichen und halben Anführungszeichen	551
15.4 Satzzeichen, Auslassungs- und Fortsetzungspunkte	554
15.4.1 Schreibweise von Satzzeichen	554
15.5 Verschiedene Sonderzeichen, E-Mail-Adressen	554
15.6 Ligaturen	557
15.6.1 Schreibweise von Ligaturen	558
15.7 Zahlen und Ziffern	559
15.7.1 Schreibweise von Zahlen	560
15.8 Nummern	562
15.9 Striche	563
15.10 Klammern, geschachtelte Klammern	565
15.10.1 Schreibweise von Klammern	566
15.11 Wortzwischenräume, Abstände (Sparationierung)	567
15.12 Mathematischer Satz	569
15.13 Worttrennungen	572
15.14 Zeilenumbruch – <i>overflow boxes</i>	574
15.15 Spezielle Literatur	575
A Von L^AT_EX 2.09 nach L^AT_EX 2_ε	577
B Seitenlayout	579
C Dateinamenserweiterungen	581

D sample2e-Beispiel 583

E Literatur 591

Beispielindex 601

Befehlsindex 605

Autorenindex 623

Index..... 625

Tabellenverzeichnis

2.1	Deutsche und französische Anführungszeichen ..	32
3.1	L ^A T _E X-Basispakete	55
3.2	Gliederungsbefehle der Dokumentklassen	60
3.3	Nummerierung der Gliederungsebenen	62
5.1	Schriftfamilien	113
5.2	Schriftformen	114
5.3	Schriftstärke	114
5.4	Kodierungen	115
5.5	\text...-Befehle	116
5.6	Deklaration von Schriften	116
5.7	Schriftumgebungen	117
5.8	Schriftgrößen	118
5.9	Angloamerikanische Anführungszeichen	121
6.1	Griechische Buchstaben	156
6.2	Benannte binäre Operatoren	157
6.3	Relationen	159
6.4	Pfeile (spezielle Relationen)	159
6.5	Operatoren in variabler Größe	162
6.6	Vordefinierte Funktionen	163
6.7	Klammersymbole	163
6.8	Verschiedene Symbole	167
6.9	Akzente im mathematischen Modus	167
7.1	Symbole im Paket latexsym	189
8.1	babel: Sprachen und Optionen	208
8.2	Sprachoptionen für Deutsch	210
8.3	Bezeichnungen für Deutsch	212

8.4	Sprachoptionen für Englisch	216
8.5	Bezeichnungen für Englisch	217
8.6	Sprachoptionen für Französisch	219
8.7	Bezeichnungen für Französisch	220
10.1	Treiber für die Standard-Grafikpakete	319
10.2	Treiber der Standard-Grafikpakete für ältere oder ungewöhnliche Ausgabegeräte	319
10.3	Fähigkeiten spezieller Gerätetreiber	320
10.4	Fähigkeiten der allgemeinen Gerätetreiber	321
11.1	Aktuelle Pakete für PostScript-Fonts und die von ihnen eingesetzten Schriftfamilien	348
11.2	Standardfonts des psnfss-Pakets	349
11.3	Symbole im Font ‚Zapf Dingbats‘	359
11.4	Symbole im Font ‚Symbol‘	362
11.5	Veraltete Pakete für PostScript-Fonts und die von ihnen eingesetzten Schriftfamilien	365
12.1	Varianten großer griechischer Buchstaben im Italic-Font	418
12.2	Zusätzliche vordefinierte Funktionsbezeich- nungen von amsmath	424
13.1	Optionen der $\mathcal{AMS}$ -Klassen	431
13.2	Pfeile und negierte Pfeile im Paket amssymb	487
13.3	Binäre Relationen im Paket amssymb	488
13.4	Negierte binäre Relationen im Paket amssymb ...	489
13.5	Binäre Operatoren im Paket amssymb	489
13.6	Klammersymbole in Paket amssymb	490
13.7	Mathematische Symbole im Paket amssymb	490

Abbildungsverzeichnis

2.1	Grundprinzip der Anwendung	15
2.2	Ausgabe von <code>xdvi</code>	17
2.3	Ausgabe von <code>dvipdfmx</code>	19
3.1	Vermaßung einer Überschrift	64
4.1	Aufbau einer Liste	75
5.1	CM-Font und Kodierung	112
9.1	Vergleich von Fußnotenpositionierung im Standardlayout mit <code>multicol</code> und <code>ftnright</code>	258
9.2	Beispiel zum Mehrspaltensatz mit <code>multicols</code> (Seite 1)	278
9.3	Beispiel zum Mehrspaltensatz mit <code>multicols</code> (Seite 2)	281
10.1	Farbnamen im Paket <code>color</code>	324
10.2	Drehen von Textelementen bei Vorgabe des Drehpunkts mit Hilfe von <code>origin</code>	332
11.1	„Helvetica“ unterschiedlich skaliert im Vergleich zur „Times Roman“	353
13.1	Symbolfonts der $\mathcal{M}\mathcal{S}$	486
13.2	Fraktur-Font	491
13.3	Script-Font	493
13.4	Euler-Font	494
13.5	Ergänzungen zum „Euler“-Font	495
14.1	Emacs mit mehreren Puffern und geöffnetem Pulldown-Menü <code>Edit</code>	504

14.2	Emacs beim Editieren einer $\LaTeX$ -Datei	508
14.3	AUCT $\TeX$ -Menü $\LaTeX$ mit Untermenü Section	512
14.4	AUCT $\TeX$ -Menü Command mit Untermenü TeXing Options	513
14.5	T $\TeX$ maker $\LaTeX$ -Referenz	515
14.6	Eingabe von Symbolen mit grafischen Oberflächen	517
14.7	Tabellen-Editor	518
14.8	Erzeugen von Gleitobjekten	518
14.9	Quickstart beim Öffnen einer Datei	519
14.10	T $\TeX$ maker-Oberfläche	519
14.11	T $\TeX$ nicCenter-Oberfläche	520
14.12	Kile-Oberfläche	521
14.13	Ly X -Oberfläche	526
14.14	Auszug aus einem Ly X -Dokument	527
14.15	Begrüßungsdatei des T $\TeX$ macs	534
14.16	Eine aktive GNUplot-Sitzung im T $\TeX$ macs	543
B.1	Layout einer rechten/ungeraden Seite	579
B.2	Layout einer linken/geraden Seite	580

Teil I

Einstieg

Einleitung

In den letzten Jahren hat der Computer die Schreibmaschine vollständig abgelöst. Textverarbeitungsprogramme bieten heute Möglichkeiten, an die vor einigen Jahren nicht zu denken war, auch wenn sich in manchen Anleitungen, beispielsweise zum Verfassen wissenschaftlicher Arbeiten, immer noch Empfehlungen wie eineinhalbzeiliger Zeilenabstand finden (ein eindeutiges Indiz einer Schreibmaschinendenkweise). Jedoch wird beim Einsatz gängiger Textverarbeitungsprogramme häufig vergessen (auch im professionellen Bereich), dass *professioneller Textsatz* die Erfahrung und Ausbildung eines eigenen Berufsstands, des Setzers, erfordert. Schöne, perfekt nach typografischen Gesichtspunkten gesetzte Bücher, die dadurch angenehm zu lesen sind, sieht man immer seltener. Der Beruf des Setzers stirbt aus – seine Arbeit übernimmt der Computer.

Leider sind die meisten heute im Einsatz befindlichen Textverarbeitungsprogramme immer noch nicht in der Lage, das Können und die Erfahrung eines Setzers vollständig nachzuempfinden. Beispielsweise wird von vielen Textverarbeitungsprogrammen die logische Struktur eines Texts, d. h. die Gliederung nach inhaltlichen Gesichtspunkten in Kapitel, Abschnitte und Absätze, nicht wirklich gefordert (auch wenn es natürlich möglich ist). Eine wichtige Aufgabe eines guten Textsatzsystems ist es, Text und Struktur klar zu trennen und ein passendes Layout bereitzustellen. Ein weiteres Beispiel ist der Zeilenumbruch in Verbindung mit einer automatischen Silbentrennung. Hier wird oft lediglich auf Zeilenbasis gearbeitet. Die Vorgehensweise dabei ist die Folgende: Eine Zeile wird mit Text gefüllt; wenn das letzte Wort nicht mehr in die Zeile passt, dann wird entweder eine Trennstelle gesucht und getrennt, wenn eine gefunden wurde, oder das ganze Wort kommt in die nächste Zeile. Die Leerräume zwischen den Wörtern der aktuellen Zeile werden vergrößert, bis das letzte Wort am rechten Rand der Zeile steht. Damit werden die Wortzwischenräume aufeinander folgender Zeilen ungleichmäßig groß. Besser wäre es in

diesen Fällen, einen größeren Bereich – beispielsweise einen Absatz – auf einmal zu bewerten, die besten Trenn- und Zeilenumbruchstellen zu ermitteln und dann mit den optimalen Positionen den Satz vorzunehmen.

Ein Textsatzsystem, das die oben aufgeführten Probleme und viele weitere vermeidet, ist $\text{\LaTeX}$, genauer das Programm $\text{\TeX}$ und das dazugehörige Makropaket $\text{\LaTeX}$. Es nimmt Autoren den größten Teil der Layoutaufgaben ab und gibt ihnen auf diese Weise die Möglichkeit, sich ganz auf den Inhalt des Texts zu konzentrieren. Um die Möglichkeiten von $\text{\LaTeX}$ zu erweitern, sind hunderte von weiteren Makropaketen und zusätzliche Software entwickelt worden, die zu $\text{\LaTeX}$ hinzugefügt (geladen) bzw. in Verbindung mit $\text{\LaTeX}$ eingesetzt werden können.

$\text{\LaTeX}$, $\text{\TeX}$, alle Makropakete und weitere Software sind frei verfügbar und für alle wesentlichen Betriebssysteme – von Linux sowie allen anderen Unix-Systemen, über Windows, MacOS, Amiga bis zu OS/2 u. v. a. – vorhanden.

1.1 $\text{\TeX}$

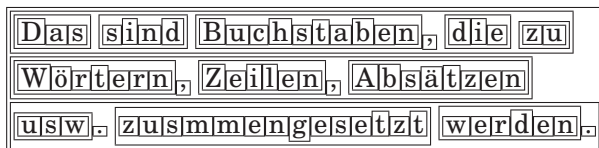
$\text{\TeX}$, gesprochen „tech“ (wie im Wort „Blech“), wurde gegen Ende der 70er Jahre von Prof. Dr. Donald E. Knuth an der Universität Stanford, Kalifornien, USA, zuerst unter dem Namen SEAL entwickelt und als $\text{\TeX}$ Anfang der 80er Jahre der Allgemeinheit zur kostenlosen Verwendung übergeben. Es ist ein ausgesprochen leistungsfähiges Satzsystem, welches die Arbeit eines Setzers (nahezu) perfekt nachbildet. Seine besondere Stärke liegt im exquisiten Satz mathematischer Formeln. Man kann sich die Arbeitsweise wie beim professionellen Handsatz mit Bleiletttern vorstellen. Möchte man beispielsweise den Text

Das sind Buchstaben, die zu
Wörtern, Zeilen, Absätzen
usw. zusammengesetzt werden.

setzen, dann legt $\text{\TeX}$ für jeden Buchstaben ein passendes Rechteck, d. h. eine Box fest, und fasst diese zu einem Wort, der nächst größeren Box zusammen. Die Worte wiederum werden zu einer Zeile, auch wieder eine Box, und diese zu einem Absatz zusammengefasst:



Um die Darstellung überschaubarer zu gestalten, so dass man hier zu Demonstrationszwecken die Boxen deutlicher sieht, werden die Zeichen sichtbar gemacht, und es wird zwischen die Boxen ein wenig Leerraum eingefügt:



T_EX, das für die griechischen Buchstaben $\tau\epsilon\chi$ steht, ist ein eingetragenes Warenzeichen (engl. *trademark*) und muss, falls das Logo nicht verfügbar ist, als TeX geschrieben werden; TEX oder tex sind nicht erlaubt. Zuerst nur gedacht, um den Band 2 von *The Art of Computer Programming* (Knuth, 1981) neu zu setzen, wurde es dann von Knuth für alle seine Bücher verwendet. Parallel dazu wurde von ihm das Programm METAFONT zur Erzeugung der Schriftzeichen (engl. *Fonts*) entwickelt. Auch wenn er immer im Gespräch betont, dass T_EX und METAFONT nur zum Satz von Büchern, genauer seiner Bücher gedacht waren, wurde schnell deutlich, dass erheblich mehr Möglichkeiten in der Konzeption dieser Software vorhanden sind. Der Algorithmus für den Zeilen- und Seitenumbruch ist unübertroffen, ebenso der Formelsatz und die fremdsprachigen Diakritika einschließlich dem Schreiben von rechts nach links (u. a. für arabische Schrift und hebräisch) sowie zeichenweise von oben nach unten (asiatische Schriften).

Aus der Sicht der elektronischen Datenverarbeitung ist T_EX eine Programmiersprache. Seine ca. 300 Befehle, *primitive* genannt, werden von T_EX verarbeitet, dem Interpreter. Seine wesentliche Stärke jedoch ist seine Makrofähigkeit, also die Möglichkeit, die *primitives* zu komplexen Befehlsfolgen zusammenzufassen. Knuth lieferte mit plain ein Makropaket, dessen Befehle von ihm in Knuth (1986e) beschrieben wurden. Die *American Mathematics Society* (AMS) stellte 1982 das Paket amstex zum Satz ihrer Publikationen zur Verfügung. Auch L^AT_EX, welches 1985 in der Version 2.09 freigegeben wurde, und seine Weiterentwicklung L^AT_EX 2_ε sind genau genommen lediglich Makropakete zu T_EX.

Die Ausgabe von T_EX ist eine sogenannte DVI-Datei (engl. *DeVice Independent*), welche den fertig formatierten Text enthält, aber völlig unabhängig vom Ausgabe-Medium ist. Diese Ausgabe wird ohne weitere Änderungen in einem zweiten Schritt von einem Druckertreiber für den entsprechenden Drucker aufbereitet. Das könnte alles von Nadel-, Tintenstrahl-, Laserdrucker bis zu einer Lichtsatanlage sein. Alternativ kann man die gleiche DVI-Datei mittels eines Bildschirmtreibers

an einem Bildschirm anzeigen. Eine weitere Möglichkeit ist, die DVI-Datei in PostScript oder PDF zu konvertieren und dann mit Programmen für diese Formate weiterzuarbeiten (beispielsweise Ghostview oder FreePDF).

Mit der Freigabe von $\text{\TeX}$ offerierte Knuth für jeden gefundenen Fehler sowohl im Programmcode als auch in der Dokumentation eine finanzielle Belohnung, die sich jedes Jahr verdoppelte. Mit jeder neuen, korrigierten Version wurden vom ihm die Schecks versendet. Dies sorgte dafür, dass viele $\text{\TeX}$ -Begeisterte sich auf die Suche nach Fehlern machten, auch wenn nach Aussage von Knuth die meisten vom ihm ausgestellten Schecks gar nicht eingelöst wurden. In jedem Fall gilt $\text{\TeX}$ als eines der fehlerfreisten Programme der letzten Jahrzehnte. Daran ändert auch nichts, dass Knuth Mitte der 90er Jahre darum bat, ihn von seiner Zusage der jährlichen Verdoppelung zu entbinden (mit dem Hinweis auf die Geschichte vom Schachbrett/Reis-Problem¹), was gerne akzeptiert wurde – auch heute würde ein gefundener Fehler \$327,68 Wert sein, allerdings sind keine weiteren wesentlichen Fehler in $\text{\TeX}$ bekannt geworden.

Es entspricht der Philosophie von $\text{\TeX}$, dass Knuth außer Fehlerbeseitigung keine wesentlichen Syntaxänderungen am System vorgenommen hat. Lediglich im Jahr 1989 auf der 10ten $\text{\TeX}$ -Konferenz in Stanford erkannte er den Wunsch der europäischen $\text{\TeX}$ -Anwender an, den Zeichensatz von 128 auf 256 Zeichen zu erweitern. Dies war notwendig geworden, um die europäischen Schriftzeichen korrekt zu verarbeiten. Im Jahr 1990 ging diese Modifikation als $\text{\TeX}$ Version 3 und als letzte große Änderung in die Verteilung. Durch diese konservative Haltung ist es jedoch möglich, in den 80er Jahren erstellte Texte heute (d. h. nach über 25 Jahren) noch korrekt zu verarbeiten. Eine Situation, von der andere Software-Produkte noch nicht einmal zu träumen wagen.

Mit $\text{\TeX}$ 3.14 erklärte Knuth jede Weiterentwicklung an $\text{\TeX}$ für beendet. Um dies zu unterstreichen, wird zukünftig die Versionsnummer immer eine weitere Dezimalstelle der Kreiszahl π erhalten und sich damit immer mehr π nähern. Im Moment liegt $\text{\TeX}$ in der Version 3.14159 vor. Professor Knuth hat sich das alleinige Recht auf den Namen $\text{\TeX}$ und die Beseitigung eventueller Fehler vorbehalten. Ähnliches gilt im Übrigen auch bei METAFONT. Hier ist die aktuelle Version 2.71828, die weite-

¹ Der Erfinder des Schachspiels forderte angeblich als Lohn ein Reiskorn, das auf jedem der 64 Felder verdoppelt werden sollte. Also auf dem ersten Feld 1 Reiskorn $= 2^0$, auf dem zweiten Feld 2 Reiskörner $= 2^1$, am Beginn der zweiten Reihe, dem neunten Feld, $2^8 = 256$. Zu Beginn der dritten Reihe, dem 17. Feld, sind es schon $2^{16} = 65\,536$ Reiskörner. Insgesamt wären es $2^{64} - 1$ Reiskörner, was noch nicht einmal durch die heutige Weltjahresernte abgedeckt würde.

ren Dezimalstellen werden sich ϵ annähern. Es ist der explizite Wunsch von Knuth, dass im Falle seines Todes die damit endgültige Version von T_EX auf die Version π und von METAFONT auf die Version e gesetzt wird.

Da der Programmcode jedoch frei verfügbar ist, konnten die Entwicklungen unter anderen Namen weiter gehen. Dazu gehört bis heute beispielsweise ϵ -T_EX, pdfT_EX, NTS (New Typesetting System), Ω (Omega), ConT_EXt und weitere Programme, die T_EX heutigen Erfordernissen anpassen. Es gibt ganz aktuelle Entwicklungen wie beispielsweise α T_EX, die ihrerseits auf neuen Entwicklungen hier von NTS aufsetzen, mit dem Ziel weiterer Verbesserungen. Was Prof. Knuth mit T_EX begann, hat sich zu einer *never ending story* entwickelt.

Für die Organisation, Unterstützung und Information der T_EX-Anwender wurde in den USA 1980 die T_EX Users Group (TUG) gegründet. Weltweit sind in den folgenden Jahren in den verschiedenen Ländern bzw. Sprachräumen Anwendervereinigungen entstanden, in denen sich Benutzer ehrenamtlich engagieren – im deutschsprachigen Raum DANTE, Deutschsprachige Anwendervereinigung T_EX e. V.

1.2 L^AT_EX

Es war Dr. Leslie Lamport, der zu Beginn der 80er Jahre ein Makropaket namens L^AT_EX, im Deutschen gesprochen „latech“, entwickelte. L^AT_EX ist ein komfortables Makropaket, das die Basisbefehle von T_EX zu Markup-Befehlen zusammenfasst. Für den Anwender wird es dadurch leichter, ein Dokument auf einfache Weise zu strukturieren, ohne auf die vielen Einzelheiten der Formatierung achten zu müssen. Er muss sich im Prinzip nur um die logische Struktur seines Texts kümmern und möglichst wenig gestalterische Details modifizieren. Bei der Erstellung eines Texts mit Hilfe von L^AT_EX wird dem Anwender die Wahl von Schriftarten, Schriftgrößen, Abständen u. Ä. abgenommen. Fußnoten, Tabellen, Register wie Inhalts-, Tabellen-, Abbildungs- und Literaturverzeichnisse werden mit geringem Aufwand erstellt. Ebenso werden Querverweise sowohl vorwärts als auch rückwärts automatisch erzeugt. Einfache Liniengrafiken sind ebenfalls möglich. Das Einbinden von extern erzeugten Bildern und Grafiken in unterschiedlichen Formaten wird unterstützt. Nicht zu vergessen ist natürlich der perfekte Satz mathematischer Formeln, der in L^AT_EX weiter vereinfacht und erweitert wurde. Die Aufzählung könnte beliebig lange fortgeführt werden.

Die Struktur des Texts wird mittels einer XML/HTML-ähnlichen Syntax bei einem vorgegebenen Layout ausgezeichnet (obwohl L^AT_EX älter ist; die

wahre Grundlage ist SGML (*Structured Generalized Markup Language*). Beispielsweise genügt das Auszeichnungsmerkmal Kapitel bzw.

```
\chapter{Dies ist eine "Überschrift"}
```

für die Formatierung der Zeichenkette „Dies ist eine Überschrift“, um eine Kapitelüberschrift mit folgenden Eigenschaften zu erzeugen:

- Entsprechende Schrift, d. h. einer größeren Schriftgröße und Schriftart je nach Layout;
- Beginn einer neuen Seite, beim Layout für Bücher außerdem auf einer rechten Seite;
- eine fortlaufende Nummer, um eins größer als das vorherige Kapitel;
- Rücksetzen aller Zähler der darunterliegenden Unterkapitel und aller weiteren Zähler für Tabellen, Grafiken und Formeln u. Ä.;
- Eintrag ins Inhaltsverzeichnis
- und einiges mehr.

Im Jahr 1985 erschien als Referenz das $\text{\LaTeX}$ -Buch von Leslie Lamport, das die Version $\text{\LaTeX}$ 2.09 beschrieb. $\text{\LaTeX}$ wurde weltweit populär. Viele wissenschaftliche Bücher sind unter Verwendung dieses Makropakets geschrieben worden, nicht nur wegen des überlegenen mathematischen Formelsatzes von $\text{\TeX}$, sondern wegen der typografischen Qualität der Ausgabe, die $\text{\LaTeX}$ erzeugte.

Mit der Anzahl der Anwender stieg auch das Interesse an einer Erweiterung von $\text{\LaTeX}$. Nach etlichen Gesprächen auf der 10ten $\text{\TeX}$ -Konferenz 1989 in Stanford, Kalifornien, USA, begann eine Gruppe um Frank Mittelbach, Chris Rowley und Dr. Rainer Schöpf in Absprache mit Leslie Lamport mit dem $\text{\LaTeX}3$ -Projekt, einer Neuimplementierung und Erweiterung von $\text{\LaTeX}$. Ziel ist dabei, das Basissystem besser zu strukturieren und mit grundlegenden Befehlen zu versehen, das dann je nach Anwendung durch Laden von Makropaketen erweitert wird. Diese Pakete stellen zusätzliche Funktionen zur Verfügung für Tabellen, beim Einbinden von Bildern, beim mathematischen Formelsatz, beim Textsatz in verschiedenen Sprachen u. v. m.

Dieses Projekt ist auf Grund seiner Komplexität auch heute noch nicht abgeschlossen. Jedoch entschloss sich das $\text{\LaTeX}3$ -Team, auch auf ständiges Drängen der Anwender, eine sehr stabile Zwischenversion unter dem Namen $\text{\LaTeX}2_\epsilon$ zu veröffentlichen. Hiermit stehen viele der Erweiterungen schon zur Verfügung. Es hat die alte Version $\text{\LaTeX}$ 2.09 mittlerweile vollständig abgelöst. Natürlich wurde auch hier die Möglichkeit geschaffen, Dokumente, die noch mit $\text{\LaTeX}$ 2.09 erstellt wurden, im sogenannten *compatibility mode* weiter zu bearbeiten.

Ende des Jahres 2003 wurde die Entwicklung von $\text{\LaTeX} 2_{\epsilon}$ für abgeschlossen erklärt, das Basissystem wird nicht mehr verändert. Das bedeutet aber nicht, dass die Erstellung von Makropaketen stagniert. Es werden immer neue, mächtigere Pakete zur Verfügung gestellt; ein Ende ist nicht abzusehen. Bleibt zu hoffen, dass irgendwann einmal der Sprung zu $\text{\LaTeX} 3$ geschafft wird, auch wenn die erste Generation der Entwickler sich heute anderen Aufgaben stellt – es kommen immer wieder junge mit neuen Ideen nach. Eine Herausforderung bleibt immer der optische Randausgleich, bei dem im üblichen Blocksatz Zeichen wie Trennstriche, Kommas und Punkte ein wenig in den Rand versetzt werden, um einen optisch perfekteren rechten Rand zu erzeugen. Dies ist eine Technik, die auf den Satz der Bibel von Gutenberg zurückgeführt werden kann. Knuth hat bereits in seinem $\text{\TeX}$ -Buch (Knuth, 1986e) einige Makros dazu vorgestellt. Vielleicht wird es doch noch einmal Wirklichkeit, dass es eine $\text{\LaTeX}$ -Version gibt, die, wie für $\text{\LaTeX}$ Version 3 versprochen (Goossens et al. (1995), S. vii) den automatischen optischen Randausgleich beherrscht. Andere Überlegungen zielen auf einen Rastersatz mit $\text{\LaTeX} 3$. Beispielsweise ist die $\text{\TeX}$ -Variante ConTeXt ein erster Schritt dorthin.

1.3 Zusätzliche Software

Eine vollständige Übersicht über alle vorhandene in Verbindung mit $\text{\LaTeX}$ verwendbare Software geben zu wollen, ist absehbar zum Scheitern verurteilt. Es gibt zu viele Produkte; es kommen ständig neue hinzu. Folglich werden wir das tun, was wir schon im Vorwort erklärt haben – wir behandeln das, was wir für am Wichtigsten halten. Dies werden sein:

- Benutzeroberflächen (u. a. Kile, $\text{\TeX} \text{nicCenter}$, LyX),
- Bildschirm- und Druckertreiber.

Das Wichtigste, anders gesagt, was man zwingend braucht, wird in diesem Band behandelt. Software-Produkte zum Erstellen von Verzeichnissen wie Sachverzeichnis, zur Bearbeitung von Literaturdatenbanken und Weiteres wird in den späteren Bänden vorgestellt.

1.4 Anwendervereinigungen und Software-Bezugsquellen

Die Anwendervereinigungen waren mangels kommerzieller Bezugskanäle von Anfang an diejenigen Organisationen, über welche Software und Informationen bezogen werden konnten. Sehr früh wurde in den USA eine Organisation namens *T_EX Users Group*, kurz TUG, ins Leben gerufen, deren Ziel es war, T_EX und seine Komponenten zu verteilen und bei der Installation und Anwendung (mittels *site coordinators*) Hilfe zu leisten. Zu Zeiten der Großrechner war dies keine einfache Aufgabe.

In Europa verbreitete sich T_EX ebenfalls sehr rasch. Schnell stellte sich heraus, dass hier ein großer Bedarf an Unterstützung notwendig wurde. Dies insbesondere, da zum einen die USA (noch ohne Internet) weit entfernt waren, und zum anderen die Besonderheiten der europäischen Sprachen der TUG nur sehr schwer zu vermitteln waren. Die Lösungen dieses Problems waren die Gründungen nicht nationaler, sondern mehr sprachlich ausgerichteter Benutzergruppen. Die erste war im französischen Sprachraum Gutenberg, als zweite folgte im deutschen Sprachraum (Deutschland, Österreich und Schweiz) DANTE, deutschsprachige Anwendervereinigung T_EX e. V. und als dritte wurde NTG, Niederlandstalige T_EX Gebruikersgroep, innerhalb eines kurzen Zeitraums ins Leben gerufen. Weitere Benutzergruppen folgten, und heute sind weltweit unzählige aktiv, um den Anwendern zu helfen und die Verbreitung von T_EX zu fördern.

1.4.1 T_EX Users Group

Die erste Anwendervereinigung, die sich bildete, war in den USA die *T_EX Users Group*:

T_EX Users Group (TUG)
 PO Box 23 11, Portland, OR 97208-2311, USA
 Tel. +1/503/223/9994
 E-Mail: office@tug.org <http://www.tug.org>

The T_EX Users Group (TUG) was founded in 1980 for educational and scientific purposes, to provide an organization for those who have an interest in typography and font design, and are users of the T_EX typesetting system invented by Donald E. Knuth.

In Providence, Rhode Island, USA (hier ist auch der Sitz der $\mathcal{AMS}$), wurde im Jahr 1980 eine *non profit, d. h. gemeinnützige*, Organisation gegründet, die sich bis heute für die weltweite Koordination der Interessen von $\text{T}_{\text{E}}\text{X}$ und den dazu gehörenden Produkten einsetzt.

Die TUG gibt eine Zeitschrift heraus, das TUGboat, und organisiert jährliche internationale Tagungen.

1.4.2 DANTE e. V.

Einige Jahre später bildeten sich auch in Europa Anwendervereinigungen, um auf die Bedürfnisse europäischer Sprachen besser eingehen zu können. Im deutschen Sprachraum ist dies

Deutschsprachige Anwendervereinigung $\text{T}_{\text{E}}\text{X}$ e. V. (DANTE e. V.)
 Postfach 10 18 40, D-69008 Heidelberg
 Tel. 06221/29766
 E-Mail: dante@dante.de <http://www.dante.de>

DANTE e. V. ist ein eingetragener gemeinnütziger Verein. Er wurde am 14. April 1989 in Heidelberg gegründet. Seine Aufgabe ist es, Anwender im deutschsprachigen Raum zu betreuen und bei Problemen zu beraten. Hierzu gehört unter anderem die Hilfe bei Benutzung und Installation der Software. Ebenso bei der Software-Verteilung mittels eigenem Software-Server (im Verbund der CTAN-Server) und bei der Verbreitung von Information über alles, was in der $\text{T}_{\text{E}}\text{X}$ -Welt geschieht, ist DANTE e. V. tätig. Entwicklungen von $\text{T}_{\text{E}}\text{X}$ -Software werden (finanziell) gefördert, koordiniert und eventuell sogar initiiert, wenn sich der Bedarf abzeichnet.

DANTE e. V. gibt seine eigene Vereinszeitschrift „Die $\text{T}_{\text{E}}\text{X}$ nische Komödie“ viermal im Jahr heraus, moderiert Diskussionslisten, finanziert FTP- und WWW-Server, erstellt und verteilt Software kostenlos an seine Mitglieder auf CD und DVD, organisiert Tagungen und regelmäßige $\text{T}_{\text{E}}\text{X}$ -Stammtische in vielen deutschen Städten.

Eine Mitgliedschaft ist in jedem Fall anzuraten.

1.4.3 Software-Verteilung

Die Software-Verteilung übernahm in der Vergangenheit zuerst die TUG. Später kamen die nationalen Anwendervereinigungen hinzu.

Als gemeinsame Aktion wurde das Comprehensive T_EX Archive Network (CTAN) ins Leben gerufen. In Kooperation mit dem damaligen DAN-TE e. V.-Archiv in Stuttgart, dem UKT_EX-Archiv in Aston und dem Archiv der Sam Houston State University (Huntsville, Texas, USA) wurde 1993 das größte Internet-Software-Archiv unter der Leitung von George D. Greenwade geplant. Das Ziel von CTAN war – und ist auch heute noch – jegliche T_EX-Software (Programme, Makros, Werkzeuge, Treiber, vorgefertigte Installationen für die verschiedenen Betriebssysteme usw.) weltweit zur Verfügung zu stellen. Neben den drei Zentralservern gibt es eine große Anzahl von Rechnern, die die Server spiegeln. Auf allen steht die Software zum Herunterladen in der gleichen Dateibaum-Struktur zur Verfügung. Ein ganzes Team von Freiwilligen kümmert sich darum, dass das alles reibungslos läuft.

Das weltweite Comprehensive T_EX Archive Network ist über folgende Adressen erreichbar:

<http://www.dante.de/software/ctan/>
<ftp://ftp.dante.de/tex-archive/>

Eine Übersicht über die verfügbare Software kann man online im T_EX-Katalog von Graham Williams finden:

<http://www.dante.de/tex-archive/help/Catalogue/catalogue.html>

Erste Schritte

Ziel dieses Kapitels ist es, eine Übersicht über die Möglichkeiten von $\text{\LaTeX}$ zu geben. Natürlich wird dabei nicht auf alle Optionen der Befehle eingegangen, dies wird in den folgenden Kapiteln geschehen. Nichtsdestotrotz sollte es möglich sein, schon mit diesen wenigen Informationen ein ansprechendes Dokument zu erstellen.

Aus der Sicht der elektronischen Datenverarbeitung ist $\text{\LaTeX}$ eine Programmiersprache. Der Text und die Anweisungen zur Formatierung werden gemeinsam kodiert, d. h. in eine Datei geschrieben.

2.1 Ein erstes Beispiel

Jedes $\text{\LaTeX}$ -Dokument genügt einem bestimmten Aufbau, der je nach Anwendung mehr oder weniger erweitert wird. Das folgende Beispiel zeigt dies:

```
1  % Das erste Beispiel
2  \documentclass{article}
3  \begin{document}
4    Das ist der kurze Text des ersten Beispiels.
5  \end{document}
```

- Zeile 1 ist optional und stellt lediglich einen Kommentar dar. Erkennbar ist dies durch das %-Zeichen am Anfang der Zeile.
- In Zeile 2 wird die Dokumentklasse festgelegt. In diesem Fall wird die Klasse `article` benutzt. Es gibt noch weitere Dokumentklassen (`report`, `book`, usw.), auf welche später eingegangen wird.

- Die Zeilen 3 und 5 begrenzen das Dokument, mit `\begin{document}` wird es eingeleitet, mit `\end{document}` wird es beendet. Ein auf Zeile 5 noch folgender Text wird ignoriert.
- Der eigentliche Text des Dokuments beschränkt sich hier auf eine Zeile, die Zeile 4.

Was in diesem Beispiel nur eine Zeile ist, könnten natürlich ebenso 2 000 Zeilen sein, oder gar ein ganzes Buch. Die Ausgabe, d. h. unsere einzelne Zeile, sieht wie folgt aus:

Das ist der kurze Text des ersten Beispiels.

2.2 Bearbeitung eines Dokuments

Die Bearbeitung geschieht in der Regel in drei Schritten. Im ersten wird mittels eines Texteditors die Eingabe erstellt und in einer Datei gespeichert. Diese Datei wird dann durch $\text{\LaTeX}$ bearbeitet; das ist der zweite Schritt. Ist das Dokument einwandfrei, kann das Ergebnis des $\text{\LaTeX}$ -Laufs an einem Bildschirm, Drucker oder Ähnlichem ausgegeben werden.

- (1) Mit einem Editor wird die Eingabe erstellt. Da $\text{\LaTeX}$ eine Markup-Sprache (engl. *markup language*) ist, enthält die Eingabe neben dem zu setzenden Text auch die Anweisungen zur Formatierung, also die $\text{\LaTeX}$ -Befehle, von denen einige wenige schon vorgestellt wurden.

Die Eingabe wird in einer Textdatei gespeichert, die nichts anderes enthält, außer den Zeichen, die Sie eingegeben haben. Würde man den Text mit einer gängigen Textverarbeitungssoftware schreiben, müsste man beim Speichern `Nur Text` oder ähnliches auswählen, damit keine zusätzlichen Informationen der Software gespeichert werden.

Jedes Betriebssystem stellt mindestens einen einfachen Editor zur Verfügung, mit dem man die $\text{\LaTeX}$ -Eingabe vornehmen kann. Allerdings sind diese meist wenig komfortabel. Es gibt für die hier erläuterten drei Schritte zur Bearbeitung eines Dokuments sogenannte Oberflächen, mit denen man diese recht komfortabel durchführen kann (vgl. Abschn. 14, S. 499).

- (2) Die Eingabedatei wird durch $\text{\LaTeX}$ bearbeitet, genauer mit $\text{\TeX}$ und dem darin geladenen $\text{\LaTeX}_{2\epsilon}$ -Makropaket, man nennt dies auch $\text{\LaTeX}$ -Format. Die Ausgabe ist die sogenannte DVI-Datei, wobei DVI die Abkürzung für das englische *device independent* (deutsch: geräteunabhängig) ist. Diese DVI-Datei enthält alle Informationen, die man benötigt, um sie mittels eines geeigneten Treiber-Programms (engl. *driver*) zu drucken oder anzusehen.
- (3) Dieser Schritt, also das Anschauen der DVI-Datei auf einem Bildschirm oder ein Ausdruck auf einem Drucker, erfordert einen Treiber, der sie entsprechend aufbereitet. In jeder $\text{\TeX}$ -Installation sind etliche Treiber enthalten, um die verschiedenen Ausgabegeräte zu bedienen.

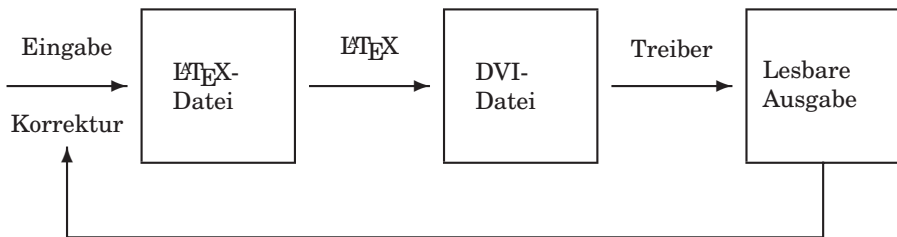


Abbildung 2.1: Grundprinzip der Anwendung

Die Grafik skizziert lediglich den Ablauf, ohne auf weitere verwendete Dateien einzugehen.

Hat man die Eingabe des ersten Beispiels von Seite 13 unter dem Namen `ErstesBeispiel.tex` gespeichert, dann wird diese Datei wie folgt bearbeitet:

```
latex ErstesBeispiel
```

Diesen Befehl kann man unter Unix beispielsweise in einer Konsole (als Terminal Programm oder auch Eingabeaufforderung bezeichnet) eingeben. Wenn man eine der später beschriebenen grafischen Oberflächen verwendet (vgl. Abschn. 14, S. 499), so gibt es dafür ein Symbol bzw. ein Menü, mit dem man diesen Befehl implizit erzeugt.

Als Ausgabe erhält man folgendes Protokoll am Bildschirm:

```
This is pdfTeX, Version 3.141592-1.30.4-2.2 (Web2C 7.5.5)
 %&-line parsing enabled.
entering extended mode
(./ErstesBeispiel.tex
```

```

LaTeX2e <2003/12/01>
Babel <v3.8g> and hyphenation patterns for english,
    usenglishmax, dumylang, nohyphenation, basque, german,
    ngerman, french, ukenglish, loaded.
(/usr/local/TeX/texmf-dist/tex/latex/base/article.cls
Document Class: article 2004/02/16 v1.4f
    Standard LaTeX document class
(/usr/local/TeX/texmf-dist/tex/latex/base/size10.clo))
No file ErstesBeispiel.aux.
[1] (./ErstesBeispiel.aux) )
Output written on ErstesBeispiel.dvi (1 page, 264 bytes).
Transcript written on ErstesBeispiel.log.

```

Dies ist eine Kurzfassung der Protokoll-Datei (.log-Datei), die unter dem Namen `ErstesBeispiel.log` erstellt wird und die folgende ausführlichen Informationen enthält (der Abdruck ist verkleinert):

```

This is pdfTeX, Version 3.141592-1.30.4-2.2 (Web2C 7.5.5)
  (format=latex 2005.12.14)  5 JAN 2006 23:07
entering extended mode
  %&-line parsing enabled.
**ErstesBeispiel
(./ErstesBeispiel.tex
LaTeX2e <2003/12/01>
Babel <v3.8g> and hyphenation patterns for english, usenglishmax,
dumylang, nohyphenation, basque, german, ngerman, french,
ukenglish, loaded.
(/usr/local/TeX/texmf-dist/tex/latex/base/article.cls
Document Class: article 2004/02/16 v1.4f Standard LaTeX document class
(/usr/local/TeX/texmf-dist/tex/latex/base/size10.clo
File: size10.clo 2004/02/16 v1.4f Standard LaTeX file (size option)
)
\c@part=\count79
\c@section=\count80
\c@subsection=\count81
\c@subsubsection=\count82
\c@paragraph=\count83
\c@subparagraph=\count84
\c@figure=\count85
\c@table=\count86
\abovecaptionskip=\skip41
\belowcaptionskip=\skip42
\bibindent=\dimen102
)
No file ErstesBeispiel.aux.
\openout1 = 'ErstesBeispiel.aux'.

LaTeX Font Info:  Checking defaults for OML/cmm/m/it on input line 3.
LaTeX Font Info:  ... okay on input line 3.
LaTeX Font Info:  Checking defaults for T1/cmr/m/n on input line 3.
LaTeX Font Info:  ... okay on input line 3.
LaTeX Font Info:  Checking defaults for OT1/cmr/m/n on input line 3.
LaTeX Font Info:  ... okay on input line 3.
LaTeX Font Info:  Checking defaults for OMS/cmsy/m/n on input line 3.
LaTeX Font Info:  ... okay on input line 3.
LaTeX Font Info:  Checking defaults for OMX/cmex/m/n on input line 3.
LaTeX Font Info:  ... okay on input line 3.
LaTeX Font Info:  Checking defaults for U/cmr/m/n on input line 3.
LaTeX Font Info:  ... okay on input line 3.
[1

```

```

] (./ErstesBeispiel.aux) )
Here is how much of TeX's memory you used:
 198 strings out of 95178
 2143 string characters out of 1184722
 45916 words of memory out of 1500000
 3429 multiletter control sequences out of 10000+50000
 3640 words of font info for 14 fonts, out of 1000000 for 2000
 36 hyphenation exceptions out of 8191
 23i,4n,17p,134b,107s stack positions out of 5000i,500n,6000p,200000b,5000s

```

Output written on ErstesBeispiel.dvi (1 page, 264 bytes).

Während der Bearbeitung sind keine Fehler aufgetreten. Man kann sowohl der Bildschirmausgabe als auch der Protokoll-Datei entnehmen, dass eine DVI-Datei unter dem Namen `ErstesBeispiel.dvi` erzeugt wurde.

Diese DVI-Datei kann man sich unter Unix mit dem Bildschirmtreiber `xdvi` anzeigen lassen. Das Kommando lautet

```
xdvi ErstesBeispiel
```

und erzeugt ein neues Fenster:



Abbildung 2.2: Ausgabe von `xdvi`

Bei Benutzung einer grafischen Oberflächen (vgl. Abschn. 14, S. 499), gibt es dafür ein Symbol bzw. ein Menü, mit dem man diesen Befehl implizit aufruft. Ein Bildschirmtreiber unter Windows ist beispielsweise YAP. Die Erzeugung alternativer Formate wie PostScript oder PDF wird im nächsten Abschnitt besprochen.

Alternative Bearbeitung eines Dokuments

Alternativ kann man aus der DVI-Datei eine PDF-Datei erstellen. PDF steht für Portable Document Format und ist heute ein weit verbreitetes Format zum Dokumentenaustausch. Es gibt hierfür mehrere Befehle – wir entschieden uns unter Unix für

```
dvipdfmx ErstesBeispiel
```

das ist das am Weitesten entwickelte Kommando, mit dem Ergebnis:

```
ErstesBeispiel.dvi -> ErstesBeispiel.pdf
[1]
4182 bytes written
```

Die entstandene PDF-Datei kann man sich anschauen, in unserem Fall wieder unter Linux (KDE-Oberfläche) mit dem Kommando:

```
kpdf ErstesBeispiel.pdf
```

was ein neues Fenster erzeugt, zu sehen in Abbildung 2.3.

Unter Windows gibt es sowohl freie PDF-Betrachter als auch den allseits bekannten Acrobat Reader der Firma Adobe Inc.

Eine weitere Alternative stellt das Ausgabeformat PostScript dar (Dateinamensendung `.ps`). Eine DVI-Datei kann in das PostScript-Format umgewandelt und dann betrachtet oder ausgedruckt werden. Das Kommando dazu lautet:

```
dvips ErstesBeispiel
```

Möchte man prüfen, ob die eigene $\text{\LaTeX}$ -Installation korrekt arbeitet, ist ein einfacher Test die Datei

```
sample2e.tex
```

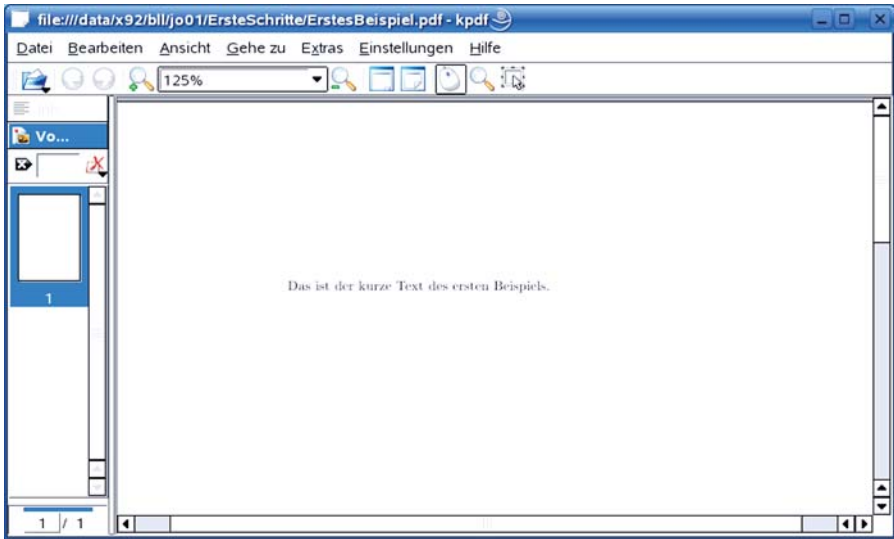


Abbildung 2.3: Ausgabe von dvipdfmx

ein dreiseitiger Beispielttext, der seit 1994 in jeder $\text{\LaTeX}$ -Distribution vorhanden ist. Sie zeigt auch schön einige Fähigkeiten von $\text{\LaTeX}$ auf, so dass sich der Vergleich von Eingabe und Ergebnis lohnt, wie im Anhang zu sehen.

Im Laufe der Zeit wurde $\text{\TeX}$ so erweitert, dass man das Format PDF auch direkt erstellen kann, ohne zuerst eine DVI-Datei zu erzeugen. Dazu gibt es eine eigene $\text{\TeX}$ -Version, sie heißt $\text{\PDF\TeX}$. Der Vorteil dieser Version liegt nicht in der gewöhnlichen Buch- oder Artikel-Produktion, die auf Papier ausgedruckt wird, sondern in der Produktion von Online-Versionen. Viele Möglichkeiten, die PDF bietet, sind mit speziellen Befehlen möglich, beispielsweise aktive Hyperlinks (beim Auswählen des Links wird die zugehörige Webseite in einem Internet-Browser geöffnet). Eine genaue Beschreibung der Kommandos folgt in einem späteren Band dieses Werks.

2.2.1 Befehle

Es gibt mehrere Arten von Befehlen:

- Eine Zeichenkette aus Buchstaben, die mit einem $\backslash$ (rückwärtiger Schrägstrich, engl. *backslash*) beginnt. Jeder Nicht-Buchstabe beendet einen solchen Befehl (Bsp. `\documentclass`).

- Ein \ (Backslash), dem ein Nicht-Buchstabe folgt (Bsp. \#).
- Ein einzelnes Sonderzeichen (Bsp. \$).

Alle Befehle können Argumente haben: Zum einen zwingend erforderliche Argumente, zum anderen optionale Argumente, die also entfallen können.

```
\<Befehlsname>
\<Befehlsname>{<Erford-Arg>}
\<Befehlsname>[<Opt-Arg>]{<Erford-Arg>}
```

- Optionale Argumente stehen in der Regel in eckigen Klammern [].
- Erforderliche Argumente werden in geschweifte Klammern eingeschlossen { }.

```
\documentclass[12pt]{article}
```

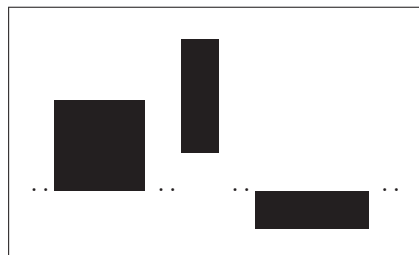
Wird mehr als ein optionales Argument angegeben, werden diese durch Komma getrennt in den eckigen Klammern aufgelistet. Wird gar kein optionales Argument verwendet, entfallen auch die eckigen Klammern.

Der Befehl

```
\rule[<Verschiebung>]{<Breite>}{<Höhe>}
```

erzeugt eine gefüllte Box, d. h. ein schwarzes Rechteck mit der Breite *<Breite>* und der Höhe *<Höhe>*. Bei Bedarf kann sie um den Wert *<Verschiebung>* von der Grundlinie verschoben werden; bei einer positiven Zahl nach oben, bei einer negativer Zahl nach unten.

```
6 \dotfill
7 \rule{1.2cm}{1.2cm}
8 \dotfill
9 \rule[0.5cm]{0.5cm}{1.5cm}
10 \dotfill
11 \rule[-0.5cm]{1.5cm}{0.5cm}
12 \dotfill
```



[Buch1/ErsteSchritte/rule.tex]

Die gepunktete Linie gibt die Position der Grundlinie an. Sie wird durch den Befehl `\dotfill` erzeugt.

Wie schon erwähnt, beendet ein Nicht-Buchstabe einen Befehl. Meist geschieht das, wie beim Befehl `\dotfill`, durch ein Leerzeichen (engl. *blank*). Hierbei muss beachtet werden, dass das auf einen Befehl folgende Leerzeichen beim Verarbeiten des Befehls entfernt wird. Auch wenn man zwei oder noch mehr Leerzeichen einsetzt, wird das Problem nicht gelöst, da mehrere Leerzeichen in der Eingabe immer zusammengefasst und zu einem Leerzeichen reduziert werden. Man kann das leicht nachvollziehen, wenn man den Befehl zum Erzeugen des $\text{\LaTeX}$ -Logos anschaut: `\LaTeX`.

```
6 Das ist ein \LaTeX Text.\
7 Das ist ein \LaTeX Text.\
8 Das ist ein \LaTeX Text.
```

Das ist ein $\text{\LaTeX}$ Text.
Das ist ein $\text{\LaTeX}$ Text.
Das ist ein $\text{\LaTeX}$ Text.

[Buch1/ErsteSchritte/latexlogo.tex]

Wie man sieht, geht das Leerzeichen zwischen dem Wort $\text{\LaTeX}$ und dem Wort Text verloren. Der Befehl `\` beginnt hier eine neue Zeile. Es gibt mehrere Möglichkeiten, dieses Problem zu lösen.

- Anfügen einer leeren Gruppe, also `{}` am Ende des Befehls;
- Einschließen des Befehls in eine Gruppe, also `{\<Befehl>}`;
- Anfügen eines *expliziten* Leerzeichens fester Breite (`\_`) nach dem Befehl;
- Falls es möglich ist, nach dem Befehl mit einem Nicht-Leerzeichen fortfahren.

In der Eingabe sieht das so aus:

```
6 Das ist ein \LaTeX{} Text.\
7 Das ist ein {\LaTeX} Text.\
8 Das ist ein \LaTeX\_ Text.\
9 Das ist ein \LaTeX-Text.
```

Das ist ein $\text{\LaTeX}$ Text.
Das ist ein $\text{\LaTeX}$ Text.
Das ist ein $\text{\LaTeX}$ Text.
Das ist ein $\text{\LaTeX}$ -Text.

[Buch1/ErsteSchritte/latexlogokorr.tex]

Variante vier ist natürlich nicht immer möglich. Die dritte sollte man vermeiden, da ein fester, also unflexibler, Abstand zwischen zwei Worten den Randausgleich erschwert. Die beiden ersten Varianten sind gleichwertig, wobei die erste den Text leichter lesbar macht und daher vorzuziehen ist.

2.2.1.1 Eigene Befehle

Es ist in $\text{\LaTeX}$ möglich, neben den vordefinierten Befehlen auch eigene Befehle zu definieren. Ohne auf alles einzugehen (das wird später nachgeholt), lautet die Syntax:

```
\newcommand{\<Befehlsname>}{<Definition>}
```

- wobei `\<Befehlsname>` für den neuen Befehlsnamen steht, mit dem der Befehl dann in der Eingabe verwendet werden kann;
- und `<Definition>` ist das, was bei jeder Verwendung des neuen Befehls im Dokument eingesetzt wird.

```
6 \newcommand{\TUG}{TUG, \TeX{} Users Group}
7 \newcommand{\DANTE}{DANTE, Deutschsprachige
8   Anwendervereinigung \TeX{} e.V.}
9 Die \TUG{}, und \DANTE{}, sind zwei Anwendervereinigungen.
```

Die TUG, $\text{\TeX}$ Users Group, und DANTE, Deutschsprachige Anwendervereinigung $\text{\TeX}$ e.V., sind zwei Anwendervereinigungen.

[Buch1/ErsteSchritte/newcommand.tex]

Wenn man Argumente verwendet, dann kann man an einen neuen Befehl, man nennt ihn auch Makro, Werte übergeben. Das geht so:

```
\newcommand{\<Befehl>}[<ArgAnz>]{<Definition>}
```

Durch das optionale Argument `<ArgAnz>` wird festgelegt, dass bei Angabe einer Zahl zwischen 1 und 9 dem neuen Befehl ein bis neun Argumente übergeben werden können.

```
6 \newcommand{\meins}[2]{Das ist das erste Argument: #1,
7   und hier kommt der zweite Wert: #2.}
8 \meins{eins}{zwei}
9 \meins{A}{B}
```

Das ist das erste Argument: eins, und hier kommt der zweite Wert: zwei.
Das ist das erste Argument: A, und hier kommt der zweite Wert: B.

[Buch1/ErsteSchritte/newcommand2.tex]

Wichtig ist, dass die neuen Befehlsnamen wirklich neu sind und noch nicht existieren. $\LaTeX$ prüft dies nach und gibt bei Verstoß eine Fehlermeldung aus.

Natürlich ist es auch möglich, bestehende Befehle zu überschreiben bzw. umzudefinieren. Das muss aber durch einen abgewandelten `\newcommand`-Befehl geschehen:

```
\renewcommand{\<Befehl>}{<Definition>}
\renewcommand{\<Befehl>}[<ArgAnz>]{<Definition>}
```

Hier gilt die Bedingung, dass der Befehl schon vorhanden sein muss. Ist dies nicht der Fall, wird auch hier ein Fehler generiert. Wir werden diese Konstruktion später des Öfteren benötigen.

2.2.2 Umgebungen

Umgebungen werden eingerichtet, um den in ihnen stehenden Text entsprechend ihrer Definition zu formatieren. Hierfür können Bearbeitungsmerkmale, beispielsweise Schriftart, Seitenbreite und vieles mehr temporär für die Umgebung geändert werden.

```
\begin{<Umgebungsname>}
  <Text in der Umgebung>
\end{<Umgebungsname>}
```

Beispielsweise bewirkt die Umgebung `quote`, dass der Text innerhalb der Umgebung auf der linken und rechten Seite gegenüber dem übrigen Text eingerückt wird. Gleichzeitig kann für die Dauer der Umgebung die Schriftart mit `\bfseries` nach fett gewechselt werden:

```
8  Etwas Text vor der Umgebung, damit man die
9  Einrückung auch wirklich optisch sieht.
10 \begin{quote}
11   \bfseries Eine Umgebung, die begonnen
12   wurde, muss auch beendet werden.
13 \end{quote}
14 Und nun noch etwas Text nach der Umgebung.
```

Etwas Text vor der Umgebung, damit man die Einrückung auch wirklich optisch sieht.

Eine Umgebung, die begonnen wurde, muss auch beendet werden.

Und nun noch etwas Text nach der Umgebung.

[Buch1/ErsteSchritte/quote.tex]

Umgebungen dürfen geschachtelt werden, man muss nur darauf achten, dass die innere Umgebung auch immer zuerst beendet wird.

Wie bei den Befehlen ist es auch möglich, eigene Umgebungen anzulegen.

```
\newenvironment{<Name>}{<BegDef>}{<EndDef>}
\newenvironment{<Name>}[<ArgAnz>]{<BegDef>}{<EndDef>}
```

- wobei *<Name>* für den neuen Umgebungsnamen steht, mit dem die Umgebung dann aufgerufen werden kann;
- *<BegDef>* ist die Anfangs-Definition, d. h. die Befehle, die bei jedem Aufruf der Umgebung zuerst ausgeführt werden;
- *<EndDef>* ist die Ende-Definition, d. h. die Befehle, die beim Beenden der Umgebung ausgeführt werden.

Damit kann man gleich für das Beispiel von eben eine eigene Umgebung `bfquote` definieren, die den Text fett druckt und wie eine `quote`-Umgebung einrückt.

```
8 \newenvironment{bfquote}{\begin{quote}
9   \bfseries}{\end{quote}}
10 Etwas Text vor der Umgebung, damit man die
11 Einrückung auch wirklich optisch sieht.
12 \begin{bfquote}
13   Dies ist ein Text mit Fettschrift auf
14   beiden Seite eingerückt.
15 \end{bfquote}
16 Und nun noch etwas Text nach der Umgebung.
```

Etwas Text vor der Umgebung, damit man die Einrückung auch wirklich optisch sieht.

Dies ist ein Text mit Fettschrift auf beiden Seite eingerückt.

Und nun noch etwas Text nach der Umgebung.

[Buch1/ErsteSchritte/bfquote.tex]

Wieder kann analog zum Befehl `\newcommand` auch hier durch das optionale Argument `<ArgAnz>`, dessen Wert zwischen 0 und 9 liegen muss, die Anzahl der Argumente vereinbart werden, die beim Aufruf übergeben werden sollen. Innerhalb der Definition `<BegDef>` werden diese Argumente als #1 bis #9 verfügbar gemacht. Beim Verlassen der Umgebung sind keine Argumente möglich.

Mit den Befehlen `\itshape` und `\scshape` kann ein Text in Kursivschrift und in Kapitälchen (kleinen Großbuchstaben) gesetzt werden. Möchte man verschiedene Schriften mit der `quote`-Umgebung kombinieren, ist dazu ein Argument notwendig:

```

8  \newenvironment{xxquote}[1]{\begin{quote}
9    #1}{\end{quote}}
10 \begin{xxquote}{\itshape}
11   Hier ein Text in Kursivschrift.
12   Hier ein Text in Kursivschrift.
13   Hier ein Text in Kursivschrift.
14 \end{xxquote}
15 \begin{xxquote}{\scshape}
16   Hier ein Text in Kapitälchen.
17   Hier ein Text in Kapitälchen.
18   Hier ein Text in Kapitälchen.
19 \end{xxquote}
```

*Hier ein Text in Kursivschrift. Hier ein Text in Kursivschrift.
Hier ein Text in Kursivschrift.*

HIER EIN TEXT IN KAPITÄLCHEN. HIER EIN TEXT IN KAPITÄLCHEN. HIER EIN TEXT IN KAPITÄLCHEN.

[Buch1/ErsteSchritte/xxquote.tex]

In Zeile 10 wird die Umgebung `xxquote` mit dem Argument `\itshape` aufgerufen und in Zeile 14 beendet. Das zweite Mal wird die Umgebung

`xxquote` in Zeile 15 mit dem Argument `\scshape` aufgerufen und in Zeile 19 beendet.

Der Vollständigkeit halber sei noch erwähnt, dass die Umgebung vor ihrer Definition nicht existieren darf. Andernfalls muss man die folgende Variante verwenden:

```
\renewenvironment{<Name>}{<BegDef>}{<EndDef>}
\renewenvironment{<Name>}[<ArgAnz>]{<BegDef>}{<EndDef>}
```

Hier gilt analog, dass die Umgebung bereits existieren muss.

Die alte Definition eines Befehls oder einer Umgebung ist nach der Neudefinition mit `\renewcommand` bzw. `\renewenvironment` nicht mehr verfügbar.

2.2.3 Gruppen

Mit Hilfe von geschweiften Klammern

```
{ }
```

können Textteile zu einer Einheit zusammengefasst werden. Solch eine Einheit bezeichnet man als einen *Bereich*, einen *Block* oder eine *Gruppe*. Wenn innerhalb einer Gruppe Formatierungsvorschriften geändert werden, so gilt beim Verlassen wieder der vorherige Zustand.

```
8 Dies ist ein Absatz,
9 {\itshape in welchem
10 einige Worte} in kursiver
11 Schrift gesetzt werden.
```

Dies ist ein Absatz, *in welchem einige Worte* in kursiver Schrift gesetzt werden.

[Buch1/ErsteSchritte/gruppe.tex]

Innerhalb der Gruppe werden in diesem Fall alle Worte kursiv gesetzt, danach wird automatisch die vorherige Schrift verwendet.

Eine Umgebung bewirkt ein ähnliches Verhalten. Auch hier können die Regeln zur Formatierung geändert werden, beim Verlassen gilt wieder der vorherige Zustand. Gruppen sind die erste Wahl für kürzere Textteile. Umgebungen sind dagegen auch für längere Textteile vorgesehen, die sich über mehrere Absätze erstrecken können.

2.2.4 Kommentare

In einem $\text{\LaTeX}$ -Dokument können Kommentare verwendet werden. Dies dient dazu, zusätzliche Bemerkungen in die Eingabe des Texts zu schreiben, die in der Ausgabe nicht sichtbar sind. Das `%`-Zeichen leitet Kommentare ein.

`%`

Alles, was nach einem Kommentarzeichen in derselben Zeile folgt, wird ignoriert, also nicht in die Ausgabe übernommen.

```
8 % Beispiel eines Kommentars
9 Ein Kommentar. % sieht man
```

Ein Kommentar.

[Buch1/ErsteSchritte/kommentar.tex]

Aus diesem Beispiel folgt: Steht das `%`-Zeichen in der ersten Spalte, dann ist die gesamte Zeile Kommentar und wird ignoriert. Steht das `%`-Zeichen irgendwo in der Zeile, dann ist nur der Rest dieser Zeile Kommentar.

Es gibt eine weitere wichtige Verwendung des `%`-Zeichens.

```
8 % Noch ein Kommentar
9 Hier ist ein Beispiel-
10 Text mit Leer-%
11 Zeichen am Zeilen-%
12 Ende.
```

Hier ist ein Beispiel- Text mit Leer-
Zeichen am Zeilen-Ende.

[Buch1/ErsteSchritte/kommentar2.tex]

Hier ist die erste Zeile, wie oben beschrieben, Kommentar. Die Zeilen 10 und 11 beinhalten jeweils am Zeilenende ein mit Bindestrich zusammengesetztes Wort. Hier kommt zum Tragen, dass am Ende jeder Zeile prinzipiell ein Leerzeichen hinzugefügt wird. Dies ist notwendig, damit beim Zeilenwechsel das Leerzeichen zwischen den Worten nicht vergessen wird. Dies kann man deutlich am Wort ‚Beispiel-Text‘ (Eingabe Zeile 9) sehen, zwischen dem hier fälschlicherweise ein Leerzeichen eingefügt wird. Beim Wort ‚Zeilen-Ende‘ kann man sehen, wie mit einem `%`-Zeichen das automatisch angefügte Leerzeichen entfällt. Durch die Kennzeichnung mit einem Kommentar nach dem Bindestrich wird das Leerzeichen entfernt. Diese Technik ist beim Erstellen eigener Befehle oft sehr wichtig.

2.3 Struktur eines Dokuments

Die Struktur eines Dokuments ist in $\text{\LaTeX}$ vorgegeben. Es gibt ein Grundgerüst (das Layout), das für jedes $\text{\LaTeX}$ -Dokument vorgegeben ist.

```
\documentclass[<Option(en)>]{<Layout>}
% Präambel
...
\begin{document}
% Text des Dokuments
...
\end{document}
```

2.3.1 Texteingabe

Zwischen den Befehlen `\begin{document}` und `\end{document}`, d. h. in der `document`-Umgebung, wird der Text des Dokuments aufgeführt. Buchstaben werden zu Wörtern, Wörter zu Sätzen und Sätze zu Absätzen zusammengefügt. Ein Absatz (engl. *paragraph*) ist eine inhaltliche Struktur im Text, er stellt einen eigenen Gedanken dar.

Für $\text{\LaTeX}$ ist das jeweilige Ende einer dieser Einheiten erkennbar:

- das Ende eines Wortes durch ein oder mehrere Leerzeichen;
- das Ende eines Satzes am Satzendezeichen, beispielsweise einem Punkt;
- das Ende eines Absatzes durch eine Leerzeile oder den Befehl `\par`.

Darüber hinaus gilt:

- Leerzeichen am Zeilenanfang werden bis zum ersten Nicht-Leerzeichen ignoriert. Das bietet die Möglichkeit, die Eingabe besser zu strukturieren.
- Aufeinanderfolgende Leerzeichen werden immer auf ein Leerzeichen reduziert.
- Am Ende jeder Zeile wird ein Leerzeichen eingefügt (s. 2.2.4, S. 27).

Was auf der Ebene von Leerzeichen gilt, ist auch für die Ebene der Leerzeilen gültig:

- Mehrere Leerzeilen werden auf eine Leerzeile reduziert.

Das bedingt, dass für horizontalen und vertikalen Leerraum spezielle $\text{\LaTeX}$ -Befehle verwendet werden müssen (s. 5.4.2, S. 131).

Wörter dürfen vom Autor eines Texts bei der Eingabe nicht getrennt werden, da dies automatisch vom Trennalgorithmus erledigt wird.

$\text{\LaTeX}$ bearbeitet bei der Formatierung der Texteingabe immer ganze Absätze. Auch wenn man einen Befehl kennen lernt, mit dem man eine neue Zeile beginnen kann, dann ist dies kein Absatz im eigentlichen Sinn. Eine Leerzeile ist die übersichtlichste Methode, einen neuen Absatz zu beginnen. Würde man einen Text mit riesigen Absätzen oder gar keiner Absatzstruktur verfassen, so hätte $\text{\TeX}$ daran schwer zu arbeiten, man hätte lange Laufzeiten bei der Bearbeitung. In Extremfällen könnte der Formatierungsprozess abbrechen.

Als Zeichen für die Eingabe stehen in der Regel alle Zeichen einer Standardtastatur zur Verfügung:

- Buchstaben: a, b, c, ..., z; A, B, C, ..., Z;
- Ziffern: 0, 1, 2, ..., 9;
- Sonderzeichen: . : ; , ? ! \ ' " + - @ & \$
% # / * = _ ^ ~ () [] < > { } | \

Bei den einfachen Anführungszeichen unterscheidet man ein linkes (accent aigu) und ein rechtes (accent grave) Anführungszeichen. Das rechte befindet sich auf einer deutschen PC-Tastatur über dem Nummern-Zeichen (#). Das linke Anführungszeichen muss unter Windows in einem einfachen Texteditor mit einem nachfolgenden Leerzeichen eingegeben werden, da sonst der Akzent über dem ersten Buchstaben des nachfolgenden Worts sitzt. Benutzt man dagegen eine grafische Oberfläche, so entscheidet diese in der Regel von sich aus, ob ein Anfangs- oder Ende-Anführungszeichen eingesetzt werden soll.

Ferner sind einige Sonderzeichen mit Steuerfunktionen belegt, d. h. sie sind *aktiv* und werden von $\text{\LaTeX}$ als Befehle erkannt. Die aktiven Zeichen sind:

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

Hiervon ist das letzte Zeichen, der Backslash, das Wichtigste, da mit ihm alle Befehle (außer den aktiven Sonderzeichen) beginnen.

Möchte man im Text diese Sonderzeichen eingeben, so ist ein Backslash (\) voranzustellen.

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

Möchte man den Backslash selbst schreiben, muss er in der Eingabe durch den Befehl `\textbackslash` kodiert werden; das einfachere `\\` hat eine andere Bedeutung.

```
\textbackslash
```

Darüberhinaus ist es möglich, die Zeichen

```
" | < >
```

in L^AT_EX für spezielle Zwecke zu verwenden. Im Text kann man für diese Zeichen die Befehle

```
\dq \textbar \textless \textgreater
```

schreiben.

2.3.1.1 Umlaute und Eszett

Die deutsche Sprache beinhaltet sieben Sonderzeichen, die zunächst in T_EX nicht enthalten waren. Es handelt sich hierbei um die Zeichen ä, Ä, ö, Ö, ü, Ü, d. h. die Umlaute, und das ß (Eszett, manchmal auch als scharfes s bezeichnet). Um diese Zeichen zu schreiben, gibt es mehrere Möglichkeiten:

- Eine Möglichkeit ist es, mit dem Befehl `"<Buchstabe>` ein Trema auf die Buchstaben a, A, o, O, u, U zu setzen. Für das Eszett steht der Befehl `\ss` zu Verfügung.

```
6 \{"a} \{"A}
7 \{"o} \{"O}
8 \{"u} \{"U}
9 \ss{}
10 \end{document}
```

ä Ä ö Ö ü Ü ß

[Buch1/ErsteSchritte/umlaut.tex]

Nachteil dieses Verfahrens ist, dass es sich dabei eigentlich um Akzente handelt. Damit sind die Pünktchen über den Buchstaben zu hoch positioniert. Die Eingabe des Eszett durch `\ss{}` ist kompliziert.

- Schon früh wurde das Makropaket `german` von Dr. Hubert Partl und anderen konzipiert, das später von Bernd Raichle perfektioniert wurde.

```
\usepackage{german}
```

Das Paket für die neue deutsche Rechtschreibung heißt `ngerman`.

```
\usepackage{ngerman}
```

Beide stellen unter vielen anderen die Befehle `"a`, `"A`, `"o`, `"O`, `"u`, `"U`, `"s` bereit, mit denen Umlaute und Eszett mit korrektem Satz (die Pünktchen sitzen auf der richtigen Höhe – beim `Ö` sieht man es besonders gut) verwendet werden können.

```
5 \usepackage{ngerman}
6 \begin{document}
7 "a "A "o "O "u "U "s
8 \end{document}
```

ä Ä ö Ö ü Ü ß

[Buch1/ErsteSchritte/umlautger.tex]

- Noch bequemer ist natürlich die Eingabe der Umlaute mit den entsprechenden Tasten auf einer deutsch angepassten Tastatur. Dies erfordert die Verwendung des Makropakets `inputenc`.

```
\usepackage[<Kodierung>]{inputenc}
```

Für das Argument `<Kodierung>` muss die benutzte Kodierung angegeben werden. Dies ist bei heute aktuellen europäischen Betriebssystemen unter

- Linux die Kodierung `utf8`,
- Windows die Kodierung `latin1`.
- MacIntosh die Kodierung `applemac`.

```
7 \usepackage[utf8]{inputenc}
8 \begin{document}
9 ä Ä ö Ö ü Ü ß
10 \end{document}
```

ä Ä ö Ö ü Ü ß

[Buch1/ErsteSchritte/umlautinp.tex]

Es wird eine weit größere Anzahl von Kodierungen unterstützt.

2.3.2 Dokumentklassen

Die Dokumentklasse bestimmt das *Layout* des Texts. Sie wird, wie bereits erwähnt, durch den Befehl

```
\documentclass[<Option(en)>]{<Layout>}
```

am Anfang der $\text{\LaTeX}$ -Eingabe festgelegt. Standardmäßig beinhaltet $\text{\LaTeX}$ folgende Dokumentklassen:

article	letter
report	slides
book	

Es stehen noch weitere Klassen zur Verfügung, dazu später mehr.

An $\langle\text{Option(en)}\rangle$ können für den Moment folgende Werte angegeben werden:

10pt	Standard, kann aber angegeben werden
11pt	
12pt	

Mit dieser Option wird die Grundschriftgröße festgelegt. Damit werden in Abhängigkeit alle weiteren Maße festgelegt. Eine andere interessante Option ist

draft

Mit dieser Option werden sogenannte *overflow boxes* durch ein schwarzes Rechteck am rechten Seitenrand sichtbar gemacht, d. h. Stellen, bei denen $\text{\LaTeX}$ Probleme hatte, eine gute Stelle für den Zeilenumbruch zu finden.

Als letzte von vielen weiteren Optionen soll in diesem einführenden Kapitel die im europäischen Sprachraum wichtige Option

a4paper

vorgestellt werden, da hiermit die Vermaßung für die Papiergröße A4, d. h. 210 mm $\times$ 297 mm festgelegt wird.

Alle gewünschten Optionen werden aufgelistet und durch Komma getrennt. Es dürfen keine Leerzeichen eingefügt werden.

Im Anschluss an den Befehl `\documentclass` folgt die sogenannte *Präambel*, der Bereich für weitere Definitionen, wie beispielsweise Texthöhe oder Textbreite. Mit dem Befehl

```
\setlength{\parindent}{0cm}
```

kann z. B. die Absatzeinrückung im gesamten Dokument abgeschaltet werden.¹

Ebenso ist es sinnvoll, in der Präambel weitere Makropakete zu laden mit dem Befehl

```
\usepackage[<Option(en)>]{<Paket>}
```

Beispielsweise das Paket mit den deutschsprachigen Anpassungen

```
\usepackage{ngerman}
```

Das folgende Beispiel zeigt ein vollständiges Dokument mit den oben beschriebenen Befehlen und Optionen.

```
1 \documentclass[12pt,a4paper]{article}
2 \usepackage{ngerman}
3 \setlength{\textwidth}{5cm}
4 \setlength{\parindent}{0cm}
5 \pagestyle{empty}
6 \begin{document}
7   Hier steht der Text
8   des Dokuments.
9
10  Und hier noch ein klein
11  wenig mehr Text.
12 \end{document}
```

Hier steht der Text des Dokuments.

Und hier noch ein klein wenig mehr Text.

[Buch1/ErsteSchritte/dklassen.tex]

2.3.3 Seitenstil

Neben dem Layout kann für jedes Dokument der Aufbau der Seite definiert werden. Hierbei wird festgelegt, ob eine Kopfzeile bzw. eine Fußzeile gesetzt wird und wie sie aufgebaut sein sollen.

¹ Dies ist in jedem Fall sinnvoller, als im ganzen Dokument mit `\\` neue Zeilen zu beginnen, um die Absatzeinrückung zu vermeiden (was man immer mal wieder sieht). Noch besser ist die Verwendung des Pakets `parskip` oder `gar` der Dokumentklassen aus KOMA-Script.

```
\pagestyle{<Layout>}
```

Standardmäßig wird das Layout `plain` verwendet. Hier ist der Seitenkopf leer, und im Seitenfuß steht zentriert die Seitennummer.

Eine andere Möglichkeit wäre das Layout `empty`. Hier bleiben Seitenkopf und -fuß leer. Dieses Layout wird bei allen Beispielen in diesem Buch verwendet.

2.3.4 Zeilen- und Seitenumbruch

Sich um den Zeilenumbruch zu sorgen, ist zunächst nicht notwendig. $\text{\LaTeX}$ hat einen ausgezeichneten Trennalgorithmus. Trotzdem ist es nicht zu vermeiden, dass einmal nicht die geeignete Stelle zur Trennung zu finden ist. $\text{\LaTeX}$ generiert dann eine Warnung der Art:

```
Overfull \hbox ...
```

In den meisten Fällen bedeutet das, dass die Ursache eruiert werden muss. Der Grund, warum keine Trennstelle ermittelt werden kann, könnte sein, dass das Wort zusammengesetzt ist; diese Wörter sind von der Trennung ausgenommen. Das Wort enthält ein spezielles Zeichen, den Bindestrich. Alle Wörter mit Sonderzeichen oder Ziffern sind für die Trennung gesperrt. Es könnte aber auch sein, dass das betreffende Wort einfach keine Trennstelle enthält.

Man kann $\text{\LaTeX}$ unterstützen, indem man mögliche Trennstellen markiert. Der Befehl lautet:

```
\-
```

Dann wird an diesen markierten Trennstellen, und zwar ausschließlich dort, bei Bedarf getrennt. Beispielsweise sieht das wie folgt aus:

```
Zeilen\ -um\ -bruch
Trenn\ -stelle
```

Hat man das Makropaket `german` bzw. `ngerman` geladen, kann alternativ zu `\-` der Befehl

```
" -
```

verwendet werden. Hierbei bleiben zusätzlich zu den markierten Trennstellen alle vom Trennalgorithmus selbst gefundenen Trennstellen erhalten.

Zeilen"-umbruch Trenn"-vorschlagsliste

Eine dritte Möglichkeit ist eine allgemeine Trennvorschrift am Anfang des Texts. Anstatt in jedem Wort, das nicht korrekt getrennt wird, die Trennstellen vorzugeben, wird die allgemeine Trennvorschrift in die Präambel des Dokuments eingefügt.

```
\hyphenation{Trenn-vor-schlag}
```

Die Markierung der Trennstellen im Wort *<Trennvorschlag>* übernimmt der Bindestrich. Alle Trennstellen im Wort müssen markiert werden. Es kann auch eine Liste von Worten angegeben werden, durch Leerzeichen getrennt.

Mühsam ist, dass jedes Wort mit seinen Flexionen einzeln aufgenommen werden muss. Sonderzeichen in den Trennvorschlägen sind nur insoweit erlaubt, als sie auch sonst in der Eingabe verwendet werden können (vgl. Umlaute, 2.3.1.1, S. 30). Wörter mit Bindestrich können natürlich nicht durch `\hyphenation` mit Trennvorschlägen versehen werden.

Mit dem Befehl `\newline` kann eine neue Zeile begonnen werden, ohne dass bei der vorherigen Zeile ein Randausgleich gemacht wird. Alternativ kann dafür auch – da einfacher – der Befehl `\\` verwendet werden. Diese beiden Befehle beenden eine Zeile, aber keinen Absatz, auch wenn sie scheinbar funktionieren, und dürfen nicht dafür verwendet werden. Sie sind also nur innerhalb eines Absatzes einzusetzen.

Wird gegen diese Regel verstoßen, reagiert $\text{\LaTeX}$ mit einer Fehlermeldung:

```
Underfull \hbox (badness 10000)
in paragraph at lines ...
```

Eigentlich ist es nur eine Warnung; man sollte die Ursache aber beseitigen, auch wenn die Ausgabe zunächst richtig aussehen mag.

Bei sehr kurzen Zeilen, beispielsweise in schmalen Spalten, nehmen die Probleme beim Zeilenumbruch zu, und Warnungen der Art

```
Overfull/Underfull \hbox ...
```

werden immer zahlreicher. Hier bietet $\text{\LaTeX}$ den Befehl `\sloppy` (engl. *unsauber*) an, der größere Wortabstände erlaubt. Das Gegenteil bewirkt der Befehl `\fussy` (engl. *schwer zufrieden zu stellen*), der die Einstellungen von `\sloppy` wieder rückgängig macht. Allerdings sollte man

`\sloppy` nur gezielt einsetzen, schon gar nicht in der Präambel, d. h. global für den gesamten Text.

Auch der Seitenumbruch kann $\text{\LaTeX}$ überlassen werden. Möchte man jedoch manuell einen Seitenumbruch erzeugen, steht hierfür der Befehl `\newpage` zur Verfügung.

2.4 Gliederung von Texten

Bei vielen Texten ist es nützlich und notwendig, sie übersichtlich zu gliedern. Übliche Gliederungen sind Kapitel und Unterkapitel, jedoch können auch Aufzählungen und Tabellen dazu verwendet werden.

2.4.1 Kapitel und Abschnitte

Die Einteilung eines Dokuments in Kapitel und Abschnitte bzw. Unterkapitel wird von $\text{\LaTeX}$ unterstützt. Hierzu können folgende Gliederungsbefehle verwendet werden:

<code>\part</code>	<code>\chapter</code>	
<code>\section</code>	<code>\subsection</code>	<code>\subsubsection</code>
<code>\paragraph</code>	<code>\subparagraph</code>	

Die Verwendung ist für alle Befehle analog:

```
\<Gliederungsbefehl>\<Überschrifttext>
```

Beispiel:

```
\section{Sinnvoll}
```

Es wird eine hierarchische Ordnung verwendet; Kapitel, Abschnitte und Unterabschnitte werden durchnummeriert. Bei der Dokumentklasse `article` wird die Zählung erst bei `\section` begonnen. `\part` wird unabhängig von den anderen Zählungen durchgeführt. Ansonsten werden beim Aufruf einer übergeordneten Ebene (bei `\subsection` ist das `\section` oder `\part`) alle Zähler für die Nummerierung zurückgesetzt bzw. die der übergeordneten Ebene um eins erhöht.

```

9 \section{Erste Abschnitts"überschrift}
10 \subsection{Erste Unterabschnitts"überschrift}
11 \subsection{Zweite Unterabschnitts"überschrift}
12 \section{Zweite Abschnitts"überschrift}
13 \subsection{Dritte Unterabschnitts"überschrift}
14 \subsubsection{Erste Unterabschnitts"überschrift}

```

1 Erste Abschnittsüberschrift

1.1 Erste Unterabschnittsüberschrift

1.2 Zweite Unterabschnittsüberschrift

2 Zweite Abschnittsüberschrift

2.1 Dritte Unterabschnittsüberschrift

2.1.1 Erste Unterabschnittsüberschrift zweiter Ebene

[Buch1/ErsteSchritte/section.tex]

2.4.2 Inhaltsverzeichnis

Das Inhaltsverzeichnis wird von $\LaTeX$ auf Basis der Kapitel- und Abschnittsüberschriften automatisch generiert. Hierzu ist es lediglich noch notwendig, an die Stelle, an der das Inhaltsverzeichnis (engl. *table of contents*) eingefügt werden soll, den Befehl

```
\tableofcontents
```

zu schreiben. $\LaTeX$ notiert sich beim ersten Bearbeitungslauf des Texts das Inhaltsverzeichnis in einer Datei mit dem Suffix (Dateinamenserweiterung) `.toc`. Beim nächsten Durchlauf wird das Inhaltsverzeichnis dann eingebunden. Das bedeutet, dass für ein korrektes Dokument, in dem die Verweise im Inhaltsverzeichnis richtig sind, mindestens zwei $\LaTeX$ -Durchläufe notwendig sind. Dies ist allerdings keine sehr große Einschränkung, denn bis zur Fertigstellung eines Texts werden regelmäßig mehrere Programmdurchläufe nötig sein.

```

1 \documentclass{article}
2 \usepackage[ngerman]{babel}
3 % textwidth nur fuer Buch
4 \setlength{\textwidth}{5cm}
5 \setlength{\parindent}{0cm}
6 \pagestyle{empty}
7 \begin{document}
8 % Groesse nur fuer Buch
9 \footnotesize
10 \tableofcontents
11 \section{Aaa}
12 \subsection{Bbb}
13 \subsection{Ccc}
14 \section{Ddd}
15 \subsection{Eee}
16 \subsubsection{Fff}

```

Inhaltsverzeichnis

1	Aaa	1
1.1	Bbb	1
1.2	Ccc	1
2	Ddd	1
2.1	Eee	1
2.1.1	Fff	1
1	Aaa	
1.1	Bbb	
1.2	Ccc	
2	Ddd	
2.1	Eee	
2.1.1	Fff	

[Buch1/ErsteSchritte/tcont.tex]

2.4.3 Aufzählungen

Eine weitere Form der Gliederung ist die Aufzählung. Man erkennt sie meist durch eine Auszeichnung oder Markierung am linken Seitenrand. $\text{\LaTeX}$ kennt standardmäßig drei Aufzählungen:

- `enumerate`
- `itemize`
- `description`

Sie unterscheiden sich lediglich durch die Art der Auszeichnung. Für alle gemeinsam gilt, dass jeder Punkt der Aufzählung mit dem Befehl `\item` eingeleitet wird.

Die Aufzählungen werden in $\text{\LaTeX}$ durch Umgebungen realisiert. Sie können beliebig geschachtelt werden.

2.4.3.1 enumerate-Umgebung

Sie stellt eine nummerierte Aufzählung dar. Die Markierung der ersten Ebene sind arabische Ziffern. Die weiteren Ebenen werden zu einem späteren Zeitpunkt behandelt (s. Abschn. 4.1.1, S. 71).

```
\begin{enumerate}
  \item <Text>
  ...
\end{enumerate}
```

Der `<Text>` kann beliebig lang sein, sowohl mehrere Zeilen als auch Absätze sind möglich.

```
8 \begin{enumerate}
9 \item
10 Das ist der erste Eintrag.
11 \item
12 Das ist der zweite Eintrag.
13 \item
14 Das ist der dritte Eintrag.
15 \end{enumerate}
```

1. Das ist der erste Eintrag.
2. Das ist der zweite Eintrag.
3. Das ist der dritte Eintrag.

[Buch1/ErsteSchritte/enum.tex]

2.4.3.2 itemize-Umgebung

Bei dieser Umgebung werden die Einträge der Aufzählung auf der ersten Ebene durch einen dicken Punkt markiert. Weitere Ebenen werden zu einem späteren Zeitpunkt erläutert (s. 4.1.2, S. 72).

```
\begin{itemize}
  \item <Text>
  ...
\end{itemize}
```

Wie bei der `enumerate`-Umgebung kann der `<Text>` beliebig lang sein.

```
8 \begin{itemize}
9 \item
10 Das ist der erste Eintrag.
11 \item
12 Das ist der zweite Eintrag.
13 \item
14 Das ist der dritte Eintrag.
15 \end{itemize}
```

- Das ist der erste Eintrag.
- Das ist der zweite Eintrag.
- Das ist der dritte Eintrag.

[Buch1/ErsteSchritte/item.tex]

2.4.3.3 description-Umgebung

Hier besteht die Markierung aus einer Beschreibung, die für jeden Aufzählungspunkt als Argument eingegeben werden muss. Diese Beschreibung wird in fester Schrift vorangestellt. Für die Umgebung `description` ist nur eine Ebene vorgesehen.

```
\begin{description}
  \item[<Beschreibung>] <Text>
  ...
\end{description}
```

Wie bei den anderen Aufzählungs-Umgebungen darf der `<Text>` beliebig lang sein.

```
8 \begin{description}
9 \item[Eins:]
10 Das ist der erste Eintrag.
11 \item[Zwei:]
12 Das ist der zweite Eintrag.
13 \item[Drei:]
14 Das ist der dritte Eintrag.
15 \end{description}
```

Eins: Das ist der erste Eintrag.

Zwei: Das ist der zweite Eintrag.

Drei: Das ist der dritte Eintrag.

[Buch1/ErsteSchritte/descr.tex]

2.4.4 Tabellen

Möchte man Text spaltenweise anordnen, dann bietet $\text{\LaTeX}$ zwei Methoden an. Die eine kennt man aus der Zeit der Schreibmaschine, d. h. man arbeitet mit Tabulatoren. Die andere Methode erzeugt eine Tabelle, indem $\text{\LaTeX}$ die Breite der Spalten berechnet und mit diesen Werten die Tabelle aufbaut.

2.4.4.1 tabbing-Umgebung – Tabulatoren

Mit Tabulatoren werden, wie bei einer Schreibmaschine, Tabulatorstopps festgelegt, auf die bei der Texteingabe vorgerückt wird. Es wird dazu die Umgebung `tabbing` verwendet. Die Tabulatorstopps werden mit dem Befehl `\=` festgelegt. Es können beliebig viele (je nach Länge der Zeile) Stopps definiert werden, auf die mit dem Befehl `\>` vorgerückt wird. Das Ende jeder Zeile wird durch den Befehl `\>` bestimmt. Ein Beispiel soll dies verdeutlichen:

```

8 \begin{tabbing}
9 SpalteAA \= SpalteAB \= SpalteAC \= SpalteAD \= SpalteAE \\
10 SpalteBA \= SpalteBB \= SpalteBC \= SpalteBD \= SpalteBE \\
11 SpalteCA \= SpalteCB \= SpalteCC \= SpalteCD \= SpalteCE
12 \end{tabbing}

```

SpalteAA	SpalteAB	SpalteAC	SpalteAD	SpalteAE
SpalteBA	SpalteBB	SpalteBC	SpalteBD	SpalteBE
SpalteCA	SpalteCB	SpalteCC	SpalteCD	SpalteCE

[Buch1/ErsteSchritte/tabb1.tex]

Wenn die Spalten gleich breit sind, gibt es keinerlei Probleme. Wenn allerdings die Spalten unterschiedliche Breite haben, kann Folgendes passieren:

```

8 \begin{tabbing}
9 eins \= zwei \= drei \= vier \= f"unf \\
10 sechs \> sieben \> acht \> neun \> zehn \\
11 elf \> zw"olf \> dreizehn \> vierzehn \> f"unfzehn
12 \end{tabbing}

```

eins	zwei	drei	vier	fünf
sechs	sieben	acht	neun	zehn
elf	zwölf	dreizehn	vierzehn	fünfzehn

[Buch1/ErsteSchritte/tabb2.tex]

Hier sieht man deutlich, dass **LaTeX** konsequent zum nächsten Tabulatorstopp rückt, notfalls wird auch wieder ein Stück zurückgesprungen, um den nächsten Stopp zu erreichen.

Dieses Problem kann man dadurch umgehen, dass man eine künstliche Tabellenzeile baut, in der aus jeder Spalte der längste Eintrag genommen wird, um die Tabulatorstopps zu definieren. Da diese Zeile natürlich nicht gesetzt werden soll, wird sie mittels des Befehls `\kill` entfernt.

```

8 \begin{tabbing}
9 sechs \= sieben \= dreizehn \= vierzehn \= f"unfzehn \kill
10 eins \> zwei \> drei \> vier \> f"unf \\
11 sechs \> sieben \> acht \> neun \> zehn \\
12 elf \> zw"olf \> dreizehn \> vierzehn \> f"unfzehn
13 \end{tabbing}

```

eins	zwei	drei	vier	fünf
sechs	sieben	acht	neun	zehn
elf	zwölf	dreizehn	vierzehn	fünfzehn

[Buch1/ErsteSchritte/tabb3.tex]

2.4.4.2 tabular-Umgebung

Bei dieser Umgebung erwartet $\text{\LaTeX}$ eine Information, wie die Spalten der Tabelle zu setzen sind. Die benötigte Spaltenbreite bestimmt $\text{\LaTeX}$ selbständig aus den Zelleinträgen, d. h. dem Text.

```
\begin{tabular}{<Vereinbarung>}
...
\end{tabular}
```

Das Argument *<Vereinbarung>* gibt an, wie die Spalten gesetzt werden sollen:

<i><Vereinbarung></i>	Ausrichtung des Spaltentexts
l	linksbündig
r	rechtsbündig
c	zentriert

Zur Trennung der Spalten in einer Tabellenzeile wird das aktive Zeichen `&` verwendet.

Ein Beispiel soll das eben Erläuterte verdeutlichen:

```
8 \begin{tabular}{llcrr}
9 eins & zwei & drei & vier & fünf \\
10 sechs & sieben & acht & neun & zehn \\
11 elf & zwölf & dreizehn & vierzehn & fünfzehn \\
12 \end{tabular}
```

eins	zwei	drei	vier	fünf
sechs	sieben	acht	neun	zehn
elf	zwölf	dreizehn	vierzehn	fünfzehn

[Buch1/ErsteSchritte/tabul.tex]

Der Inhalt einer Zelle in der `tabular`-Umgebung kann nicht umbrochen werden, der Inhalt ist also auf die Seitenbreite beschränkt. Darüber hinaus kann innerhalb einer `tabular`-Umgebung kein Seitenumbruch erfolgen. Natürlich gibt es Möglichkeiten, diese Beschränkungen zu umgehen; diese werden jedoch später behandelt.

Die `tabular`-Umgebung erzeugt keinen vertikalen Abstand. Das bedeutet, dass man entweder einen Befehl für den vertikalen Leerraum verwenden muss (s. 5.4.2, S. 131) oder eine Umgebung, die Abstand erzeugt. Vorgestellt wurde schon die Umgebung `quote`; später lernt man noch weitere Umgebungen kennen, die beispielsweise zentrieren.

```

8 Text vor der Tabelle. Damit man den mit \texttt{quote}
9 erzeugten Abstand sieht.
10 \begin{quote}
11 \begin{tabular}{llcrr}
12 eins & zwei & drei & vier & f"ünf \\
13 sechs & sieben & acht & neun & zehn \\
14 elf & zw"ölf & dreizehn & vierzehn & f"ünfzehn
15 \end{tabular}
16 \end{quote}
17 Und etwas Text nach der \texttt{tabular}"=Umgebung.
```

Text vor der Tabelle. Damit man den mit `quote` erzeugten Abstand sieht.

eins	zwei	drei	vier	fünf
sechs	sieben	acht	neun	zehn
elf	zwölf	dreizehn	vierzehn	fünfzehn

Und etwas Text nach der `tabular`-Umgebung.

[Buch1/ErsteSchritte/tabulquote.tex]

2.5 Formelsatz

In der Textverarbeitung ist der Satz von mathematischen Formeln eine der schwierigsten Tätigkeiten. Bereits beim Bleisatz war das Setzen von Formeln etwas, das Spezialisten vorbehalten war. Wie schon erwähnt, war der Formelsatz ein Grund für die Entwicklung von $\text{\TeX}$, da viele Programme schon scheitern, sobald die Anforderungen das Setzen eines Bruchs übersteigen.

Der Satz von Formeln wird in einem eigenen Bearbeitungsmodus vorgenommen. Hierbei wird unterschieden, ob eine Formel im fortlaufenden

Text (Fließtext) verwendet wird, oder ob es sich um eine abgesetzte Formel handelt.

2.5.1 Formeln im Text

Mathematische Ausdrücke, die innerhalb eines Absatzes stehen, werden zwischen `$` und `$` oder `\(` und `\)` oder `\begin{math}` und `\end{math}` eingeschlossen. Welche der drei Methoden man wählt, ist ohne Bedeutung, da sie gleichwertig sind.

Indizes, tiefergesetzte Zeichen, schreibt man unter Verwendung des Befehls `_` (Unterstreichungsstrich, engl. *underscore*) mit der Syntax:

```
_{<Zeichen>}
```

und Exponenten, höhergesetzte Zeichen, mit dem Befehl `^` (Zirkumflex, engl. *hat*, umgangssprachlich auch Dach genannt) mit der Syntax:

```
^{<Zeichen>}
```

Natürlich kann man auch mathematische Sonderzeichen setzen, beispielsweise griechische Buchstaben. Den Buchstaben π erhält man durch `\pi`, sofern man sich im mathematischen Modus befindet.

Mit diesen – eigentlich doch wenigen Informationen – ist es möglich, folgenden Satz zu schreiben.

```
8  Das Volumen einer Kugel
9  wird berechnet mit:
10 $V_{K} = 4/3 \pi r^{3}$.
```

Das Volumen einer Kugel wird
berechnet mit: $V_K = 4/3\pi r^3$.

[Buch1/ErsteSchritte/math1.tex]

Zu beachten ist, dass die Befehle `_`, `^` und `\pi` ausschließlich innerhalb des mathematischen Modus verwendet werden können.

2.5.2 Abgesetzte Formeln

Möchte man einen mathematischen Ausdruck freistellen, also in einer eigenen Zeile mit vertikalem Abstand und zentriert setzen, kann man ihn in `\[` und `\]` oder `\begin{displaymath}` und `\end{displaymath}` einschließen.

```

8  Das Volumen einer Kugel
9  wird berechnet mit:
10 \[V_{K} = 4/3 \pi r^3\]

```

Das Volumen einer Kugel wird berechnet mit:

$$V_K = 4/3\pi r^3$$

[Buch1/ErsteSchritte/math2.tex]

Den kleinen Schönheitsfehler, dass der Bruch $4/3$ nicht korrekt dargestellt wird, kann man durch Verwendung des Befehls `\frac` beseitigen.

```
\frac{<Zähler>}{<Nenner>}
```

```

8  Das Volumen einer Kugel
9  wird berechnet mit:
10 \[V_{K} = \frac{4}{3} \pi r^3\]
11

```

Das Volumen einer Kugel wird berechnet mit:

$$V_K = \frac{4}{3}\pi r^3$$

[Buch1/ErsteSchritte/math3.tex]

Die vollständigen Möglichkeiten des mathematischen Formelsatzes werden in Kapitel 6 ab S. 137 behandelt.

2.6 Fazit

Dieses Kapitel der ersten Schritte soll lediglich aufzeigen, was mit ganz geringen Mitteln in $\text{\LaTeX}$ machbar ist. Die weiteren Kapitel werden tiefer in die Materie einführen.

Dokumentstrukturen

$\text{\LaTeX}$ -Dokumente haben einen festgelegten Aufbau, der im vorliegenden Kapitel beschrieben wird. Das Dokumentlayout wird bestimmt und es werden die Dokumentklassen besprochen, die zum $\text{\LaTeX} 2_{\epsilon}$ -Standard gehören. Es ist möglich, weitere Makropakete zu laden, die die Standard-Formatierung erweitern. Der grundsätzliche Aufbau einer Seite wird durch das Seitenlayout festgelegt. Alle diese Einstellungen werden vor dem Befehl `\begin{document}` vorgenommen. Diesen Bereich der Eingabe bezeichnet man als Präambel.

Titelseite und Überschriften werden behandelt, sowie das Erstellen von Anhang und Inhaltsverzeichnis.

3.1 Dokumentlayout

Der Befehl `\documentclass` legt mit seinen Argumenten das Layout für das gesamte Dokument fest. Er kann mit optionalen Argumenten versehen sein, die das Layout modifizieren.

```
\documentclass[<Option(en)>]{<Layout>}
```

Für das Argument `<Layout>` wird eine Dokumentklasse gewählt. Man nennt dies ebenfalls Layout, da alle Layout-Parameter in der Dokumentklasse definiert werden. Für das Argument `<Option(en)>` kann eine Liste von Optionen durch Komma getrennt angegeben werden.

Es sind verschiedene Layouts verfügbar, man spricht hier auch von Dokumentklassen.

article	ist geeignet für die Bearbeitung von kurzen Artikeln, aufgeteilt in Abschnitte, Unterabschnitte, usw.
report	ist gedacht für größere Artikel, aufgeteilt in Kapitel, Abschnitte, Unterabschnitte, usw. Man kann auch mehrere article-Dokumente zu einem report-Dokument zusammenfassen, jedes Kapitel ist dabei ein Artikel.
book	dient zur Bearbeitung eines Buchs, beispielsweise beginnen Kapitel hier immer auf einer rechten Seite.
letter	ist, wie der Name aussagt, für das Verfassen von Briefen vorhanden. In der originalen Version ist es jedoch für Briefe in den USA gedacht und damit für die Benutzung in Europa ungeeignet. Jedoch sind etliche Makropakete vorhanden, die die deutsche Briefnorm unterstützen.
slides	zu deutsch Folien, ist eine Klasse, um ein- und mehrfarbige Overheadfolien zu erstellen. Allerdings ist heute, durch den Einsatz von Beamern, dieses Layout nicht mehr interessant. Dafür gibt es Dokumentklassen, die für die neue Technik angepasst sind, beispielsweise beamer.
proc	dient der Formatierung von Sitzungsprotokollen (engl. proceedings). Hier erfolgt die Formatierung der Seite in zwei Spalten, die Seitenbreite ist größer als bei den anderen Layouts.

Neben diesen Dokumentklassen gibt es noch einige für spezielle Anwendungen:

ltxdoc	wird benutzt, um die Dokumentation des $\text{\LaTeX}$ -Kerns und der $\text{\LaTeX}$ -Pakete zu formatieren.
ltxguide	dient der Dokumentation von $\text{\LaTeX} 2_{\epsilon}$, beispielsweise für das Handbuch „ $\text{\LaTeX}$ for Authors“ $\text{\LaTeX} 3$ Projekt Team (2001), u. a.
ltxnews	wird gebraucht, um die $\text{\LaTeX}$ Neuigkeiten (engl. <i>news</i>) zu formatieren.
minimal	enthält die Minimaldefinitionen einer Klassen-Datei. Diese Klasse wird in der Regel für die Fehlersuche verwendet.

Die American Mathematical Society ($\mathcal{AMS}$) hat zum Satz mathematischer Formeln in ihren Publikationen erweiterte Dokumentklassen entwickelt (vgl. 13.1.1, S. 430).

amsart	für die Artikel der $\mathcal{AMS}$,
---------------	---------------------------------------

amsbook	für die Bücher der $\mathcal{A}\mathcal{M}\mathcal{S}$,
amsproc	für Proceedings,
amstex	zur Dokumentation des Quellcodes der Pakete,
amslatex	um die Dokumentation der Pakete zu formatieren.

Das Hauptproblem, das bei Verwendung der bisher genannten $\text{\LaTeX}$ -Dokumentklassen auftritt, ist, dass sie für den US-englischen Sprachraum geschaffen wurden. Die Texte, die durch sie bzw. mit ihrer Hilfe produziert werden, entsprechen den Gepflogenheiten und treffen mehr den Geschmack eines Amerikaners als den eines Europäers. Markus Kohm machte sich viel Arbeit und hat eine umfangreiche Sammlung von Dokumentklassen und Paketen unter dem Namen KOMA-Script-Paket veröffentlicht. Er hat sie nicht nur als europäisch angepasstes Gegenstück zu den Standardklassen konzipiert, sondern darüber hinaus um etliche Einstellmöglichkeiten erweitert. Diese Klassen sind an den deutschen Geschmack angepasst, besser gesagt an die Regeln der Typografie im deutschsprachigen Raum.

scrartcl	für Artikel,
scrreprt	für umfangreiche Artikel und Artikelsammlungen,
scrbook	für Bücher und
scrlettr2	für Briefe.

Die wesentlichen Zusatzpakete lauten typearea für die Satzspiegelkonstruktion, scrpage2 für Kopf- und Fußzeilen, scrdate und scrtime für Wochentag und Uhrzeit, zur Lösung von Problemen mit Paketabhängigkeiten scrfile sowie für Briefadressverwaltung scraddr. Das gesamte KOMA-Script-Paket wird in Band 2 dieses Werks besprochen.

3.1.1 Dokumentoptionen

Mit dem Befehl

```
\documentclass[<Option(en)>]{<Layout>}
```

kann für das Argument *<Option(en)>* eine Liste von Optionen durch Komma getrennt angegeben werden. Es dürfen in dieser Liste keine Leerzeichen (auch nicht nach den Kommas) verwendet werden.

10pt | 11pt | 12pt

Hiermit wird die Standardschriftgröße auf 10pt, 11pt oder 12pt eingestellt. Diese Schriftgröße gilt als Basis für alle anderen Schriftgrößen, Abstände, Überschriften und sonstige Maße, die $\text{\LaTeX}$ während der Formatierung verwendet. Ohne Angabe dieses Werts wird 10pt genommen.

**letterpaper | legalpaper | executivepaper |
a4paper | b5paper**

legt die Papiergröße fest, in der das Dokument gedruckt wird.

Papiergröße	Seitenmaße
letterpaper	8,5 in $\times$ 11 in
legalpaper	8,5 in $\times$ 14 in
executivepaper	7,25 in $\times$ 10,5 in
a4paper	210 cm $\times$ 297 mm
a5paper	148 cm $\times$ 210 mm
b5paper	176 cm $\times$ 250 mm
(1 in = 72,27 pt; 1 in = 25,4 mm)	

Die Voreinstellung bei fehlender Angabe ist `letterpaper`. Es ist natürlich auch möglich, trotz Angabe einer Option die Seitenmaße nachträglich zu ändern (s. 5.4.3, S. 133).

oneside | twoside

die Ausgabe für ein- oder zweiseitigen Druck. Letzteres wird erforderlich, damit bei doppelseitiger Ausgabe die Textränder auf beiden Seiten gleich sind. Bei der Klasse `book` wird `twoside` automatisch eingestellt. Alle anderen Klassen sind standardmäßig `oneside`.

openright | openany

Mit Hilfe dieser Option wird festgelegt, ob ein neues Kapitel auf einer ungeraden und damit rechten Seite beginnen soll. Standardmäßig ist `openright` bei der Klasse `book` aktiviert. Bei Büchern ist es angenehm, dass man mit dem Daumen der rechten Hand durchblättern und schnell den Anfang eines Kapitels finden kann.

final | draft

Wenn $\text{\LaTeX}$ keinen Umbruch findet, dann werden Zeilen erzeugt, die den rechten Rand überschreiten (Overfull hboxes). Die `draft`-Option bewirkt, dass jede dieser Zeilen mit einem schwarzen Rechteck gekennzeichnet wird. Die `final`-Option, die keine Kennzeichnung vornimmt, ist die Voreinstellung.

onecolumn | twocolumn

Die Seitenformatierung geschieht in einer oder zwei Spalten (engl. *column*). Voreingestellt ist einspaltiger Satz. Der Abstand zwischen den beiden Spalten kann mit dem Befehl `\columnsep` festgelegt werden. Die Breite der senkrechten Linien, die die beiden Spalten trennt, wird mit `\columnseprule` bestimmt. Die Voreinstellung ist 0, d. h. eine unsichtbare Linie.

titlepage | notitlepage

Bei der Dokumentklasse *article* wird der Titel (vgl. 3.5, S. 56) oberhalb des nachfolgenden Texts zentriert. Bei *report* und *book* dagegen wird der Titel auf einer eigenen Seite positioniert. Mit der Option `titlepage` wird er auch bei *article* auf einer eigenen Seite gesetzt. `notitlepage` dagegen bewirkt bei den Klassen *book* und *report* einen Titel wie bei *article* ohne eigene Seite.

landscape

Diese Option bewirkt, dass die Ausgabe im Querformat dargestellt wird. Es werden dabei Seitenhöhe und -breite vertauscht. (Achtung: Wenn wirklich Querformat gedruckt werden soll, so muss man das beim Drucken erneut angeben.)

openbib

bewirkt, dass die Bibliografie in einem „offenen“ Layout dargestellt wird.

leqno

entscheidet, auf welcher Seite die Nummer von nummerierten Gleichungen steht. Mit der Angabe der Option `leqno` befindet sich die Nummer auf der linken Seite, während sie normalerweise auf der rechten Seite gesetzt wird (l steht dabei für engl. *left*, eqn für engl. *equation*).

fleqn

Standardmäßig werden abgesetzte Formeln zentriert. Die Option `fleqn` bewirkt, dass die Gleichungen links ausgerichtet werden (fl steht für engl. *float*, eqn für engl. *equation*). Über den Befehl `\mathindent` wird die Einrückung bestimmt.

3.2 Seitenlayout

Meist möchte man neben dem Text weitere Informationen auf eine Seite drucken, mindestens die Seitennummer, oft aber auch die gerade aktuelle Hauptüberschrift.

Eine Ausgabeseite besteht bei $\text{\LaTeX}$ aus Kopfzeile, Seitenrumpf und Fußzeile. Es gibt Parameter, wie beispielsweise `\textheight` oder `\textwidth`, die die Maße einer Seite definieren (diese werden an anderer Stelle behandelt, Abschn. 5.4.3, S. 133).

Mit dem Seitenlayout wird der grundsätzliche Aufbau einer Seite festgelegt:

```
\pagestyle{<layout>}
```

Bei zweiseitigem Satz, beispielsweise für Bücher, sind linke und rechte Seiten unterschiedlich. Linke Seiten haben immer eine gerade Seitennummer, auf rechten Seiten ist sie ungerade.¹

3.2.1 Deklarationen von Kopf- und Fußzeilen

Die Layout-Optionen sind:

plain

Hier ist die Kopfzeile leer, in der Fußzeile steht die Seitennummer zentriert in der Mitte. Dies ist die Voreinstellung, wenn keine Layout-Option gesetzt ist.

empty

Kopf- und Fußzeile bleiben leer.

headings

Die Fußzeile bleibt bei diesem Layout leer. Die Kopfzeile enthält die Seitenzahl sowie Kapitel- oder Abschnittsüberschrift, je nach verwendeter Dokumentklasse.

	book/report article
Einseitiger Druck	<code>\chapter \section</code>
Zweiseitiger Druck	
linke Seite	<code>\chapter \section</code>
rechte Seite	<code>\section \subsection</code>

Auf der ersten Seite steht diese Information noch nicht zur Verfügung. Wenn man dies vermeiden möchte, kann man mit der Umgebung `\titlepage` eine Titelseite als erste Seite generieren.

Die Werte für die linke bzw. rechte Kopfzeile können bei Bedarf durch `\markright` bzw. `\markboth` überschrieben werden. Mit dem Beginn eines neuen Kapitels bzw. Abschnitts werden sie jedoch wieder auf die alten Werte gesetzt.

¹ Schauen Sie mal auf die Seitennummern des Buchs, das vor Ihnen liegt.

Eine Ausnahme sind die Seiten, auf denen ein neues Kapitel beginnt. Sie werden wie `plain` gesetzt.

myheadings

verhält sich wie `headings`, jedoch wird der Seitenkopf nicht aus den Kapitel- und Abschnittsinformationen gebildet, sondern durch die Befehle `\markright` bzw. `\markboth` festgelegt:

```
\markright{<Text>}
\markboth{<Text_L>}{<Text_R>}
```

3.2.2 Seitennummerierung

Standardmäßig werden für die Seitennummern ganze Zahlen verwendet, die aus arabischen Ziffern bestehen. Es ist jedoch möglich, andere Nummerierungen zu verwenden.

```
\pagenumbering{<Numtyp>}
```

Erlaubte Angaben für `<Numtyp>` sind:

arabic

Die Seitennummer wird mit arabischen Ziffern (0, 1, ..., 9) gebildet. Das ist die Voreinstellung.

roman

Es werden kleine römische Zahlen verwendet (i, ii, iii, iv, v, ...).

Roman

Es werden große römische Zahlen verwendet (I, II, III, IV, V, ...).

alph Es werden Kleinbuchstaben verwendet (a, b, c, ..., z).

Alph Es werden Großbuchstaben verwendet (A, B, C, ..., Z).

Das Ändern der Seitennummerierung setzt den Seitenzähler auf 1.

Es gibt weitere Möglichkeiten ein verfeinertes Layout zu erzeugen. Andere Makropakete wie beispielsweise `fancyhdr` (wird erst in Band 2 behandelt), bieten zusätzliche Befehle um jeden Teil der Kopf- und Fußzeilen sowie die Nummerierungstypen zu definieren.

3.3 Makropakete

Ein Makropaket ist eine $\text{\LaTeX}$ -Datei, die zusätzliche Funktionalität zu den Möglichkeiten hinzufügt, die der $\text{\LaTeX}$ -Kernel bietet. Jedes Paket, wenn es beispielsweise auf CTAN (vgl. 1.4.3, S. 11) abgelegt ist, besteht wenigstens aus einer `.ins`-Datei (d. h. die Datei hat die Dateinamenserweiterung `.ins`) und einer `.dtx`-Datei. Bearbeitet man die erste mit $\text{\LaTeX}$, dann erhält man den Quellcode des Pakets, genauer eine oder mehrere `.sty`-Dateien. Jede `.sty`-Datei enthält den Quellcode eines $\text{\LaTeX}$ -Makropakets. Bearbeitet man die `.dtx`-Datei mit $\text{\LaTeX}$, dann erhält man die Dokumentation zum Paket.

Man kann jedoch auf die Makropakete zugreifen, die in einer Verteilung enthalten sind, beispielsweise $\text{\TeX}$ Collection 2005: DANTE e.V., TUG, Lehmanns Fachbuchhandlung (2005). Damit erspart man sich die Arbeit, die Makropakete selbst zu installieren.

Die allgemeine Struktur eines $\text{\LaTeX}$ -Dokuments sieht wie folgt aus:

```
\documentclass[<Option(en)>]{<Layout>}
% Präambel
...
\begin{document}
% Eingabe
...
\end{document}
```

Makropakete werden in der Regel in der Präambel geladen, und zwar mit dem Befehl:

```
\usepackage[<Option(en)>]{<Paket(e)>}
```

Wobei die `<Option(en)>` und `<Paket(e)>` eine Liste von Optionen bzw. Paketen sein können. Ein Beispiel:

```
\usepackage[dvipdfm]{graphicx}
```

Hier wird das Makropaket `graphicx` mit der Option `dvipdfm` geladen.

Die folgende Tabelle gibt eine Übersicht über die für Anwender wesentlichen $\text{\LaTeX}$ -Basispakete. Die anderen Pakete in Kapitel 7, die auch alle zum Standard gehören, klassifizieren die Autoren als Hilfspakete bzw. Pakete für Paketentwickler.

Tabelle 3.1: $\text{\LaTeX}$ -Basispakete

Paketname	Abschn./S.	Kurzbeschreibung
alltt	7.1	<code>verbatim</code> -Umgebung mit Interpretation von Befehlen.
amstex	12	Definiert viele weitere Befehle für mathematischen Formeln und darüber hinaus.
babel	8	Definition von Sprachabhängigen Komponenten insbesondere für Dokumente in anderen Sprachen als englisch.
color	10.2	Wird für die Verwendung von Farbe benötigt.
fontenc	7.6	Kodierung von Zeichensätzen.
graphpap	7.7	Zum Setzen von „Millimeter“-Papier. Es definiert den <i>graphics paper</i> -Befehl in der <code>picture</code> -Umgebung.
ifthen	7.8	Definiert einfache Strukturen der Programmiersprachen.
inputenc	7.9	Kodierung der Eingabe.
latexsym	7.10	Definiert zusätzliche Symbolbefehle.
makeidx	7.11.1	Stellt Befehle für Indexerstellung zur Verfügung.
showidx	7.11.2	Bewirkt die Anzeige von Indexeinträgen im Randbereich.
textcomp	7.15	Sammlung von Textsymbolen.

3.4 Zusammenfassungen

Eine Zusammenfassung eines Reports oder Artikels kann man mit der Umgebung `abstract` formatieren. Sie soll mit einem kurzen Text eine Übersicht über den Inhalt des Dokumentes geben.

```
\begin{abstract}
... % Text der Zusammenfassung
\end{abstract}
```

Dieser Zusammenfassung wird die Überschrift ‚Abstract‘ vorangestellt. Bei Verwendung des Pakets `ngerman` natürlich ‚Zusammenfassung‘. Sie wird von den Dokumentklassen unterschiedlich formatiert:

article

Es wird die Schriftgröße `\small` verwendet, der Text auf beiden Seiten eingerückt und dem nachfolgenden Text auf derselben Seite vorangestellt.

report

Der Text wird in der Standardgröße formatiert, es wird keine Einrückung vorgenommen. Die Zusammenfassung wird auf eine eigene Seite gedruckt.

book

hier ist die Umgebung `abstract` unzulässig.

Das Verhalten des Befehls `\abstract` kann mit den Dokumentoptionen `notitlepage` bzw. `titlepage` modifiziert werden. Mit der Option `titlepage` wird auch bei `article` die Zusammenfassung auf eine eigene Seite gedruckt. Im Gegensatz dazu wird mit der Option `notitlepage` bei `report` nach der Zusammenfassung auf derselben Seite fortgefahren.

3.5 Titelseite

Es sind bei $\text{\LaTeX}$ zwei Möglichkeiten vorgesehen, um eine Titelseite zu erstellen. Zum einen gibt es eine Umgebung, um einen Titelseite frei zu gestalten, zum anderen den Befehl `\maketitle`.

3.5.1 `titlepage`-Umgebung

Die Umgebung `titlepage` dient zum Erstellen einer Titelseite.

```
\begin{titlepage}
... % Text und Befehle
\end{titlepage}
```

```
9  \begin{titlepage}
10 \begin{center}
11   {\large Titelseite}
12
13   von
14
15   \bigskip
16   \begin{tabular}{l}
17     Klaus Braune           \\
18     Joachim Lammarsch und  \\
19     Marion Lammarsch
20   \end{tabular}
21 \end{center}
22 \end{titlepage}
```

Titelseite von Klaus Braune Joachim Lammarsch und Marion Lammarsch

[Buch1/DokStrukt/title.tex]

Wie man sieht hat man hier die Möglichkeit der freien Gestaltung. Mit der `center`-Umgebung (Zeilen 10, 21) werden alle Zeilen zentriert. Um die Autorennamen trotzdem untereinander ausrichten zu können, wurde

eine einspaltige `tabular`-Umgebung verwendet (Zeilen 16, 20), die dann als Ganzes zentriert wird. Der Befehl `\bigskip` (Zeile 15) schafft einen größeren Zwischenraum vor den Autorennamen.

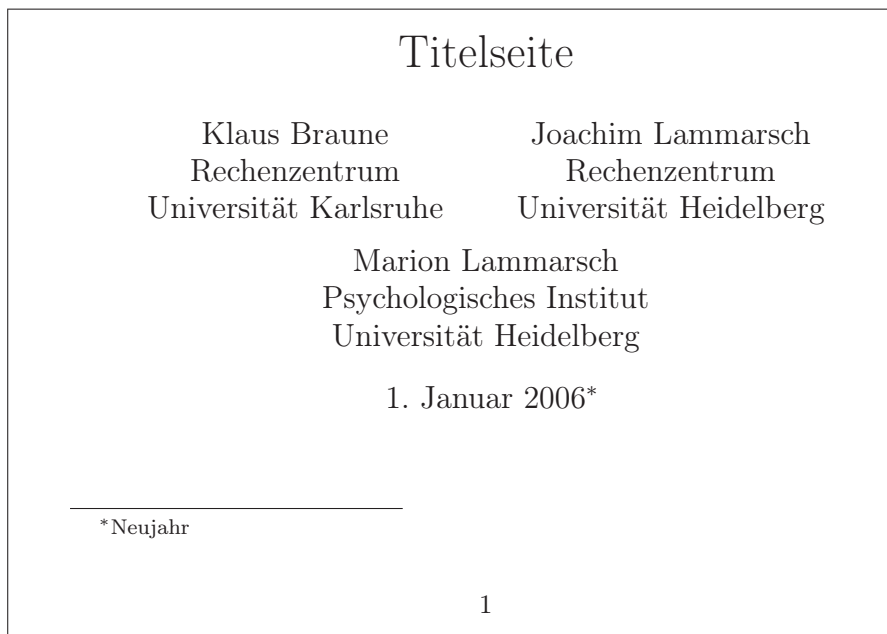
3.5.2 maketitle-Befehl

Die Titelseite wird mit dem Befehl `\maketitle` formatiert. Hierzu benötigt dieser Befehl Angaben, die mit den Befehlen `\title`, `\author`, `\date` und `\thanks` *vorher* (am besten in der Präambel) festgelegt werden. Zwingend erforderlich ist die Verwendung von `\title` und `\author`.

```
\title{<Titelüberschrift>}
\author{<Autorenname>}
```

Ist der Titel länger als eine Zeile, so wird er automatisch umbrochen; es ist jedoch auch der Befehl `\` erlaubt, der einen Zeilenvorschub erzeugt. Bei mehr als einem Author sollte man die Namen durch `\and` trennen. Standardmäßig wird das aktuelle Datum hinzugefügt. Es ist jedoch auch möglich, mittels des Befehls `\date{<Datum>}` das Datum manuell zu setzen.

```
1 \documentclass{article}
2 \usepackage[T1]{fontenc}
3 %%\usepackage[ngerman]{babel} %% Fehler mit Umlauten
4 \usepackage[ngerman]
5 \setlength{\textwidth}{11cm}
6 \setlength{\textheight}{8cm}
7 \setlength{\parindent}{0cm}
8 \pagestyle{empty}
9 \title{Titelseite}
10 \author{%
11     Klaus Braune\\Rechenzentrum\\Universit"at Karlsruhe
12     \and
13     Joachim Lammarsch\\Rechenzentrum\\
14     Universit"at Heidelberg
15     \and
16     Marion Lammarsch\\Psychologisches Institut\\
17     Universit"at Heidelberg}
18 \date{1. Januar 2006\thanks{Neujahr}}
19 \begin{document}
20 \maketitle
21 \end{document}
```



[Buch1/DokStrukt/maketitle.tex]

Hinweis: Bei diesem Beispiel wurde statt dem empfohlenen Paket `babel` mit der Option `ngerman` das Paket `ngerman` geladen.²

Datums- und Autorenangaben können bei Bedarf durch `\author{}` bzw. `\date{}` unterdrückt werden.

Zu den Parametern `\title`, `\author` und `\date` kann der Befehl

```
\thanks{<Text>}
```

hinzugefügt werden. Dadurch wird eine Markierung (vgl. 4.4.1, S. 98) eingebracht und der Text wie eine Fußnote auf der Seite gesetzt.

Bei den Dokumentklassen `report` und `book` erzeugen die `titlepage`-Umgebung und der Befehl `\maketitle` eine separate Titelseite. Im Gegensatz dazu generieren sie bei `article` nur einen Titelvorspann. Dieses Verhalten ist analog dem der `abstract`-Umgebung und kann auch hier mit den Dokumentoptionen `notitlepage` bzw. `titlepage` geändert werden.

² Der Grund liegt in einem Problem mit den mit "a kodierten Umlauten. Das `babel`-Paket arbeitet in der Präambel fehlerhaft. Schreibt man Umlaute in den Text, tritt das Problem nicht auf.

3.6 Gliederungsbefehle

Jedes noch so einfache Dokument wird aufgeteilt in Textblöcke, damit es zum einen leichter zu schreiben und zum anderen leichter zu lesen/verstehen ist. Gibt man den Textblöcken Überschriften, dann kann man dazu ein Inhaltsverzeichnis erstellen, durch welches man den Leser durch das Dokument führen kann. Das Dokument wird folglich in Abschnitte unterteilt. Jede Dokumentklasse enthält eine Anzahl Gliederungsbefehle, mit denen man dies vornehmen kann.

Die allgemeine Syntax aller Gliederungsbefehle ist:

```
\<Gliederungsbefehl>{<Überschrifttext>}
```

bzw.

```
\<Gliederungsbefehl>*{<Überschrifttext>}
```

`<Gliederungsbefehl>` steht hier für die Namen der verschiedenen Gliederungsbefehle:

<code>\part</code>	<code>\chapter</code>	
<code>\section</code>	<code>\subsection</code>	<code>\subsubsection</code>
<code>\paragraph</code>	<code>\subparagraph</code>	

Bei der ersten Variante wird eine fortlaufende Nummer hinzugefügt, bei der gesternten Variante unterbleibt die Nummerierung. Damit wird auch der interne Zähler für die Gliederungsnummer nicht geändert. Des Weiteren wird bei der ersten Variante ein Eintrag ins Inhaltsverzeichnis vorgenommen, der bei der zweiten Variante unterbleibt.

```

9  \section{Erste Abschnitts"überschrift}
10 \subsection{Erste Unterabschnitts"überschrift}
11 \subsection{Zweite Unterabschnitts"überschrift}
12 \section*{Abschnitts"überschrift ohne Nummer}
13 \subsubsection{"Überschrift 3ter Ebene}
14 \paragraph{"Überschrift 4ter Ebene}
```

1 Erste Abschnittsüberschrift

1.1 Erste Unterabschnittsüberschrift

1.2 Zweite Unterabschnittsüberschrift

Abschnittsüberschrift ohne Nummer

1.2.1 Überschrift 3ter Ebene

Überschrift 4ter Ebene

[Buch1/DokStrukt/section.tex]

Es sind folgende Gliederungsbefehle bei den verschiedenen Dokumentklassen möglich:

Tabelle 3.2: Gliederungsbefehle der Dokumentklassen

Gliederungs- Befehl	Dokumentklasse					
	article	report	book	letter	slides	proc
<code>\part</code>	+	+	+	–	–	+
<code>\chapter</code>	–	+	+	–	–	–
<code>\section</code>	+	+	+	–	–	+
<code>\subsection</code>	+	+	+	–	–	+
<code>\subsubsection</code>	+	+	+	–	–	+
<code>\paragraph</code>	+	+	+	–	–	+
<code>\subparagraph</code>	+	+	+	–	–	+

Ist im Seitenlayout der Seitenstil `headings` gewählt, wird die Überschrift von `\chapter` bzw. `\section` in den Seitenkopf übernommen. Bei den gesternten Gliederungsbefehlen unterbleibt dies, d. h. dass man eventuell keine oder eine falsche Überschrift im Seitenkopf bekommt. Ein Beispiel dafür ist in diesem Buch das Vorwort. Die Überschrift ist mit `\chapter*` gesetzt und danach mit `\markboth{Vorwort}{Vorwort}` die sogenannte ‚fortlaufende Kopfzeile‘ korrigiert.

Der Befehl `\part` hat eine Sonderstellung. Er dient dazu, ein großes Dokument in Teile zu strukturieren, hat aber keinen Einfluss auf die Nummerierung anderer Befehle. Die Zählung für `\part` erfolgt mit großen römischen Ziffern.

Das Argument der Gliederungsbefehle wird sowohl ins Inhaltsverzeichnis als auch bei entsprechendem Seitenstil in den Seitenkopf übernommen. Während im Inhaltsverzeichnis ein Zeilenumbruch noch möglich ist – oft aber nicht gewünscht, – kann der Eintrag im Seitenkopf zu lang

sein. Die Lösung für diese Problematik sind zwei Titelangaben: Eine lange Überschrift für den Text und eine Kurzüberschrift für Seitenkopf und Inhaltsverzeichnis. Aus diesem Grund haben Gliederungsbefehle noch ein optionales Argument.

```
\<Gliederungsbefehl>[<Kurztitel>][<Überschrifttext>]
```

Eine gesternte Form dieses Gliederungsbefehl gibt es natürlich nicht.

```
1 \documentclass{article}
2 \usepackage[ngerman]{babel}
3 % textwidth nur fuer Buch
4 \setlength{\textwidth}{5cm}
5 \setlength{\parindent}{0cm}
6 \pagestyle{empty}
7 \begin{document}
8 % Groesse nur fuer Buch
9 \footnotesize
10 \tableofcontents
11 \section{Aaa}
12 \subsection{Bbb}
13 \subsection[C]{Ccc}
14 \section*{Ddd}
15 \subsubsection[E]{Eee}
16 \paragraph{Fff}
17 \end{document}
```

Inhaltsverzeichnis

1	Aaa	1
1.1	Bbb	1
1.2	C	1
1.2.1	E	1

1 Aaa

1.1 Bbb

1.2 Ccc

Ddd

1.2.1 Eee

Fff

[Buch1/DokStrukt/tcont.tex]

3.6.1 Ändern der Nummerierung

Wie oben schon erwähnt, werden die Überschriften nummeriert. Dazu ist es erforderlich, dass für jeden Gliederungsbefehl ein eigener Zähler vorhanden ist. Er trägt immer den Namen des Befehls, d. h. `\chapter` verwendet den Zähler `chapter`. Dadurch kann man, falls wirklich notwendig, den Zähler modifizieren. Beispielsweise kann durch

```
\setcounter{chapter}{3}
```

der Zähler für die Kapitelüberschrift auf 3 gesetzt werden, somit wird das nächste Kapitel die Nummer 4 erhalten. Weitaus interessanter ist es jedoch, zu steuern, bis zu welcher Tiefe die Nummerierung der Gliederung durchzuführen ist. Hierzu ist jeder Gliederungsebene in Abhängigkeit von der Dokumentklasse eine Zahl zugeordnet:

Tabelle 3.3: Nummerierung der Gliederungsebenen

Gliederungsbefehl	Dokumentklasse		
	article	report	book
\part	0	-1	-1
\chapter	-	0	0
\section	1	1	1
\subsection	2	2	2
\subsubsection	3	3	3
\paragraph	4	4	4
\subparagraph	5	5	5
Voreinstellung	3	2	2

Der für jede Klasse vorgegebene Wert, d. h. die Voreinstellung kann durch den Befehl

```
\setcounter{secnumdepth}{<Wert>}
```

geändert werden, beispielsweise

```
\setcounter{secnumdepth}{5}
```

bewirkt, dass bis zur \subparagraph-Ebene nummeriert wird.

```
1 \documentclass{article}
2 \usepackage[ngerman]{babel}
3 % textwidth nur fuer Buch
4 \setlength{\textwidth}{5cm}
5 \setlength{\parindent}{0cm}
6 \pagestyle{empty}
7 \setcounter{secnumdepth}{5}
8 \begin{document}
9 % Groesse nur fuer Buch
10 \footnotesize
11 \tableofcontents
12 \section{Aaa}
13 \subsection{Bbb}
14 \subsection[C]{Ccc}
15 \section*{Ddd}
16 \subsubsection[E]{Eee}
17 \paragraph{Fff}
18 \subparagraph{Ggg}
19 \end{document}
```

Inhaltsverzeichnis

1	Aaa	1
1.1	Bbb	1
1.2	C	1
1.2.1	E	1

1	Aaa	
1.1	Bbb	
1.2	Ccc	
	Ddd	
1.2.1	Eee	
1.2.1.1	Fff	
1.2.1.1.1	Ggg	

3.6.2 Ändern von Überschrift-Ebene und Schriftart

In der Regel genügen sieben Gliederungsebenen den normalen Anforderungen. Falls jedoch eine weitere Ebene erforderlich werden sollte, dann ist $\text{\LaTeX}$ jederzeit erweiterbar. Man kann entweder Makropakete dazu verwenden, beispielsweise den Befehl `\theorem` aus den Paketen `theorem` (s. 9.19, S. 294) oder `amsthm` (s. 13.1.2, S. 433). Oder man macht eigene Anpassungen, indem man Befehle neu- bzw. redefiniert.

Auch die Art der Überschriften ist leicht zu ändern, wenn dies gewünscht wird. Außer `\part` und `\chapter`, die getrennt in jeder Dokumentklasse definiert sind, verwenden alle weiteren Gliederungsebenen den generischen Befehl `\@startsection`.

```
\@startsection{<Name>}{<Ebene>}{<Einzug>}{<Abst_V>}{<Abst_H>}{<Stil>}
```

In der Dokumentklasse `article` ist `\section` beispielsweise wie folgt definiert (aus der Datei `article.cls`):

```
1 \newcommand\section{\@startsection%
2     {section}{1}{\z@}%
3     {-3.5ex \@plus -1ex \@minus -.2ex}%
4     {2.3ex \@plus .2ex}%
5     {\normalfont\Large\bfseries}}
```

Die Argumente von `\@startsection` bedeuten hierbei:

<Name>

Der Name des Gliederungsbefehls, der definiert werden soll (ohne Backslash).

<Ebene>

Die Nummer der Ebene, der der Gliederungsbefehl zugeordnet werden soll (siehe auch Tab. 3.3).

<Einzug>

Einrückung der Überschrift gemessen von linken Rand. Ein negativer Wert verschiebt nach links, ein positiver nach rechts.

<Abst_V>

Steht für den vertikalen Abstand zwischen Überschrift und Text davor.

<Abst_H>

Steht für den vertikalen Abstand zwischen Überschrift und nachfolgendem Text.

<Stil>

Gibt an in welchem Stil (Schriftart, -größe und -form) die Überschrift gesetzt werden soll.

Die Werte von *<Einzug>*, *<Abst_V>* und *<Abst_H>* sind Längenmaße (vgl. 5.4.3).

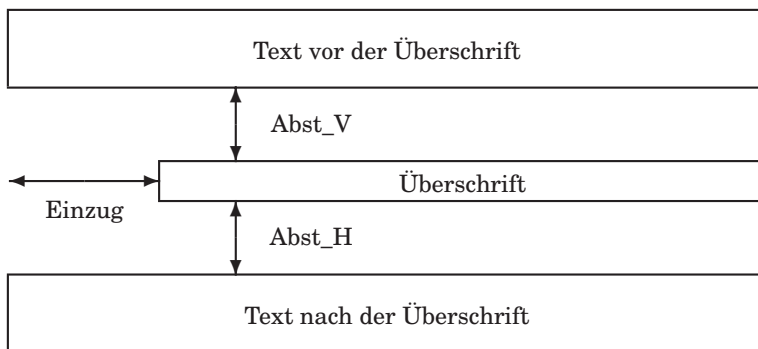


Abbildung 3.1: Vermaßung einer Überschrift

Möchte man nun die Überschrift, die der `\section`-Befehl erzeugt, nicht nur fett, sondern zusätzlich noch kursiv (italic) darstellen, geht man folgendermaßen vor:

- Man sucht entsprechend der verwendeten Dokumentklasse die Definition des `\section`-Befehls, d. h. bei `article` in `article.cls` den Code `\newcommand{section}{...}` (s. o.).
- Diese Zeilen kopiert man am Besten in eine eigene Datei mit der Namensweiterung `.sty`, beispielsweise `meinmakro.sty`.
- Die letzte Zeile dieses Makros ändert man von

```
{\normalfont\Large\bfseries}
```

zu

```
{\normalfont\Large\bfseries\itshape}
```

- Nun bindet man diese Datei in der Präambel seines eigenen Dokuments ein, mit

```
\usepackage{meinmakro}
```

- Danach hat jeder `\section`-Befehl die geänderte Schriftart.

Man hätte natürlich auch in jeden `\section`-Befehl die Schrift auf kursiv setzen können, was aber viel aufwändiger und fehleranfälliger gewesen wäre. Außerdem wäre dann auch der Eintrag im Inhaltsverzeichnis kursiv, was man in der Regel nicht möchte.

Eine weitere Gliederungsebene hinzuzufügen ist nicht viel schwerer, sondern nur ein wenig aufwändiger. Man muss sich zunächst um den zusätzlichen Zähler, die Ausgabe der Nummerierung und den Eintrag ins Inhaltsverzeichnis kümmern. Das könnte man Schritt für Schritt abarbeiten – vorhandenen Code kopieren und ändern ist einfacher.

(1) In einer Standard- $\text{\LaTeX}$ -Installation sind alle Makros vorhanden. Unter Unix liegen sie im Verzeichnis

```
.../TeX/texmf-dist/tex/latex/base
```

Der Unix-Befehl

```
grep subparagraph *
```

in diesem Verzeichnis gestartet, listet alle Zeilen (mit Angabe des Dateinamens) in denen der Begriff `subparagraph` vorkommt:

```
article.cls:\newcounter {subparagraph}[paragraph]
article.cls:\renewcommand\thesubparagraph {\theparagraph.\@arabic\c@subparagraph}
article.cls:\newcommand\subparagraph{\@startsection{subparagraph}{5}{\parindent}%
article.cls:\newcommand*\l@subparagraph{\@dottedtocline{5}{10em}{5em}}
book.cls:\newcounter {subparagraph}[paragraph]
book.cls:\renewcommand\thesubparagraph {\theparagraph.\@arabic\c@subparagraph}
book.cls:\newcommand\subparagraph{\@startsection{subparagraph}{5}{\parindent}%
book.cls:\newcommand*\l@subparagraph{\@dottedtocline{5}{12em}{6em}}
latex.ltx:\let\subparagraphmark\gobble
report.cls:\newcounter {subparagraph}[paragraph]
report.cls:\renewcommand\thesubparagraph {\theparagraph.\@arabic\c@subparagraph}
report.cls:\newcommand\subparagraph{\@startsection{subparagraph}{5}{\parindent}%
report.cls:\newcommand*\l@subparagraph{\@dottedtocline{5}{12em}{6em}}
```

Unter Windows liegen die Makros normalerweise im Verzeichnis

```
C:\Programme\...\tex\latex\base
```

Mit einer Textsuche, die man normalerweise unter Start→Suche findet, kommt man zum selben Ergebnis.

(2) Da hier mit der Dokumentklasse `article` gearbeitet werden soll, sind die Zeilen 1–4, also die Datei `article.cls`, von Interesse. Zusätzlich noch die Zeile 9, da die Datei `latex.ltx` die Basis von $\text{\LaTeX}$ ist. Kopiert man den relevanten Code aus den beiden Dateien in eine neue Datei mit dem Namen `ssubparagraph.sty`, so hat diese den Inhalt:

```

1 \let\subparagraphmark\@gobble
2 \newcounter{subparagraph}[subparagraph]
3 \renewcommand\thesubparagraph%
4   {\theparagraph.\@arabic\c@subparagraph}
5 \newcommand\subparagraph{%
6   \@startsection{subparagraph}{5}{\z@}%
7     {3.25ex \@plus1ex \@minus .2ex}%
8     {-1em}%
9     {\normalfont\normalsize\bfseries}}
10 \newcommand*\l@ssubparagraph{\@dottedtocline{6}%
11   {10em}{5em}}

```

Man hätte den Code auch zu der Datei `meinmakro` hinzufügen können.

(3) Jetzt ändert man jeden `\subparagraph`-Befehl in `\ssubparagraph`, den `\theparagraph` in `\thesubpararaph`, die Ebene von 5 nach 6, die Schrift von `\bfseries` zu `\itshape` und erhält die neue Version von `ssubparagraph.sty`:

```

1 \let\ssubparagraphmark\@gobble
2 \newcounter {ssubparagraph}[subparagraph]
3 \renewcommand\thesubparagraph%
4   {\thesubparagraph.\@arabic\c@ssubparagraph}
5 \newcommand\ssubparagraph{%
6   \@startsection{ssubparagraph}{6}{\z@}%
7     {3.25ex \@plus1ex \@minus .2ex}%
8     {-1em}%
9     {\normalfont\normalsize\itshape}}
10 \newcommand*\l@ssubparagraph{\@dottedtocline%
11   {6}{10em}{5em}}

```

Nun kann man Dokumente mit acht Gliederungsebenen schreiben.

(4) Als Letztes wird noch hinzugefügt, dass auch alle acht Ebenen im Inhaltsverzeichnis erscheinen:

```
\setcounter{tocdepth}{<Wert>}
```

Im folgenden Beispiel wird neben der eingebundenen Datei mit dem `\ssubparagraph`-Befehl die Werte von `\secnumdepth` und `\tocdepth` auf 6 gesetzt, außerdem sind die Werte bei den Einrückungen des Inhaltsverzeichnisses angepasst:

```

25 \tableofcontents
26 \section{Aaa}
27 \subsection{Bbb}
28 \subsubsection[E]{Eee}
29 \paragraph{Fff}
30 \subparagraph{Ggg}
31 \ssubparagraph{Hhh}

```

Inhaltsverzeichnis

1	Aaa	1
1.1	Bbb	1
1.1.1	E	1
1.1.1.1	Fff	1
1.1.1.1.1	Ggg	1
1.1.1.1.1.1	Hhh	1

1 Aaa

1.1 Bbb

1.1.1 Eee

1.1.1.1 Fff

1.1.1.1.1 Ggg

1.1.1.1.1.1 HhhHhh

[Buch1/DokStrukt/ssubpara.tex]

3.6.3 Spezielle Befehle für die Klasse book

Bei der Dokumentklasse book sind zur Grobstrukturierung zusätzlich weitere drei Befehle vorhanden:

```

\frontmatter
\mainmatter
\backmatter

```

Mittels `\frontmatter` wird bei der ersten Seite eines Buchs, d. h. Vorwort, Inhaltsverzeichnis, usw., die Seitenmarkierung mit kleinen römischen Ziffern vorgenommen.

Nach Aufruf des Befehls `\mainmatter` wird der Seitenzähler wieder auf 1 zurückgesetzt und auf arabische Ziffern gewechselt.

`\backmatter` schaltet für die Anhänge, Bibliografie, Schlagwortverzeichnis sowie mögliche weitere Verzeichnisse die Kapitelnummerierung ab. Die Seitenzahl wird fortgesetzt, der Stil mit arabischen Ziffern beibehalten.

3.7 Anhang

Das Setzen des Anhangs wird durch den Befehl

```
\appendix
```

begonnen. Dies bewirkt, dass die Zähler für die Gliederungsbefehle neu gesetzt werden. Die Nummerierung für die `\section`-Befehle bei `article` und die `\chapter`-Befehle bei `report` und `book` erfolgt mit Großbuchstaben. Zusätzlich wird der feststehende Text Kapitel (für andere Sprachen natürlich entsprechend, vgl. 8, S. 8) durch Anhang ersetzt.

3.8 Inhaltsverzeichnis

Es kann an jeder Stelle des Dokuments angelegt werden, indem man den Befehl

```
\tableofcontents
```

in den Text einfügt. Normalerweise geschieht dies am Anfang eines Dokuments nach dem Vorwort oder am Ende vor den Verzeichnissen wie Sachverzeichnis etc.

Die Einträge von den Gliederungsbefehlen für das Inhaltsverzeichnis werden in eine Datei mit der Namensweiterung `.toc` (*table of contents*) gespeichert. Beim nächsten Durchlauf wird diese Datei anstelle des `\tableofcontents`-Befehls eingefügt. Das bedeutet, dass (mindestens) zwei $\text{\LaTeX}$ -Durchläufe zur Erstellung des Inhaltsverzeichnisses benötigt werden.

$\text{\LaTeX}$ bietet zusätzlich die Möglichkeit, Einträge zum Inhaltsverzeichnis von Hand einzufügen. Hierzu verwendet man

```
\addcontentsline{toc}{<Ebene>}{<Text>}
```

Das Argument `<Ebene>` steuert die Formatierung des Eintrags `<Text>`. Die Seitennummer wird von der Seite übernommen, auf der der `\addcontentsline`-Befehl steht. Ein Beispiel, welches nur die Anwendung des Befehls demonstrieren soll, nicht empfehlenswert zur Realisierung:

```
\addcontentsline{toc}{3}{Neue Abschnittsüberschrift}
```

Um einen Eintrag zu formatieren, der eine Gliederungsnummer enthält, sollte der Eintrag in der Form

```
\protect\numberline{<Nummer>}{<Eintrag>}
```

vorgenommen werden.

Eine zweite Möglichkeit ist, Text oder Formatierungsbefehle direkt in das Inhaltsverzeichnis einzufügen. $\LaTeX$ bietet hier den Befehl

```
\addtocontents{toc}{<Text>}
```

an. In *<Text>* können beliebiger Text oder Formatierungsbefehle übergeben werden.

```
\addtocontents{toc}{\protect\newpage}
```

Hierdurch wird ein Seitenumbruch im Inhaltsverzeichnis erzeugt, und zwar an der Stelle, an der in der Reihenfolge der Überschriften, die ja die Einträge automatisch erzeugen, der Befehl eingefügt wurde. Die Anweisung `\protect` bewirkt, dass der darauf folgende Befehl nicht sofort ausgeführt, sondern in die `.toc`-Datei geschrieben wird. Erst beim Wiedereinfügen der `.toc`-Datei an die Stelle von `\tableofcontents` werden diese geschützten (engl. *protected*) Befehle ausgeführt.

Spezielle Textstrukturen

4.1 Aufzählungen

Hier folgt der zweite Teil der Gliederung von Texten aus dem Kapitel Erste Schritte (S. 13). Die Umgebungen `itemize` und `enumerate` werden vervollständigt und darüber hinaus erläutert, wie man eigene Listen erstellen kann.

Bei den beiden Standardlisten `enumerate` und `itemize` ist der Aufbau vorgegeben. Man kann zwischen verschiedenen Marken und Voreinstellungen wählen. Sie können bis zu vier Ebenen tief geschachtelt werden, wobei sich die Markierung jeweils ändert. Man kann die Listenarten auch mischen.

Bei den veränderbaren Listen, genannt allgemeine Listen, kann man einerseits bei der Umgebung `list` jeden Parameter zur Gestaltung der Liste verändern oder andererseits Material als Pseudoliste ohne echte Listeneinträge formatieren (`trivlist`).

4.1.1 `enumerate`-Umgebung

Die Umgebung `enumerate` liefert eine nummerierte Aufzählung, als Markierung werden arabische Ziffern, kleine Buchstaben, kleine römische Ziffern und große Buchstaben verwendet.

```
\begin{enumerate}
  \item[<Marke>] <Text>
  [\item[<Marke>] <Text>]
  ...
\end{enumerate}
```

Beim Aufruf von `\item` kann als optionales Argument (in eckigen Klammern) eine andere Markierung angeben. In diesem Fall wird der aktuelle Zähler der Listenebene nicht verändert.

```

8  Aufzählungen:
9  \begin{enumerate}
10   \item Bei \texttt{enumerate} werden die Elemente mit
11       Ziffern oder Buchstaben nummeriert.
12   \item Die Nummerierung erfolgt automatisch.
13   \item Listen kann man schachteln:
14   \begin{enumerate}
15     \item Die maximale Schachtelungstiefe
16         ist 4.
17     \item Einrückung und Bezeichnung der Elemente
18         wechseln automatisch.
19   \end{enumerate}
20   \item[X.] usw.
21 \end{enumerate}

```

Aufzählungen:

1. Bei `enumerate` werden die Elemente mit Ziffern oder Buchstaben nummeriert.
2. Die Nummerierung erfolgt automatisch.
3. Listen kann man schachteln:
 - (a) Die maximale Schachtelungstiefe ist 4.
 - (b) Einrückung und Bezeichnung der Elemente wechseln automatisch.
- X. usw.

[Buch1/SpezTextStruk/enumkomp.tex]

4.1.2 itemize-Umgebung

Die Umgebung `itemize` eignet sich für einfache Auflistungen. Für die Markierung werden die Zeichen `•`, `–`, `*` und `·` verwendet.

```

\begin{itemize}
  \item[<Marke>] <Text>
  [\item[<Marke>] <Text>]
  ...
\end{itemize}

```

Auch bei `itemize` kann im optionalen Argument `<Marke>` des Befehls `\item` eine andere Markierung angegeben werden.

```

6 \pagestyle{empty}
7 \usepackage{textcomp}
8 \begin{document}
9 Aufzählungen:
10 \begin{itemize}
11 \item Bei \texttt{itemize} werden die Elemente
12     durch Punkte und andere Symbole gekennzeichnet.
13 \item[\textcircled{P}] Die Markierung kann gesetzt werden.
14 \item Man kann auch verschiedene Listentypen schachteln:
15 \begin{itemize}
16 \item Beim Schachteln ist darauf zu achten, dass die
17     Umgebungen richtig wieder beendet werden.
18 \item Einrückung und Bezeichnung der Elemente
19     werden entsprechend dem Typ gewählt.
20 \end{itemize}
21 \item usw.
22 \end{itemize}

```

Aufzählungen:

- Bei `itemize` werden die Elemente durch Punkte und andere Symbole gekennzeichnet.
- Ⓟ Die Markierung kann gesetzt werden.
- Man kann auch verschiedene Listentypen schachteln:
 - Beim Schachteln ist darauf zu achten, dass die Umgebungen richtig wieder beendet werden.
 - Einrückung und Bezeichnung der Elemente werden entsprechend dem Typ gewählt.
- usw.

[Buch1/SpezTextStruk/itemkomp.tex]

Das Symbol in der Liste ist aus dem Paket `textcomp`.

Schachtelt man verschiedene Listentypen, kann man bis zu einer Schachteltiefe von sechs gehen. Danach erhält man die folgende Fehlermeldung von $\LaTeX$: ! Too deeply nested.

4.1.3 Allgemeine Listen

Allgemeine Listen sind Listen, deren Aussehen man selbst gestalten kann. Die Tiefe der Einrückung, den Abstand zwischen den einzelnen Absätzen, die Markierungsart der einzelnen Punkte, etc. werden definiert.

4.1.3.1 Die list-Umgebung

Die Syntax der list-Umgebung lautet:

```
\begin{list}{<MarkenDef>}{<Listendefinition>}
  \item[<Marke>] <Text>
  [\item[<Marke>] <Text>]
  ...
\end{list}
```

wobei *<MarkenDef>* die Definition der Listenmarkierung ist, die durch den `\item`-Befehl ohne optionales Argument erzeugt werden soll.

Das Argument *<Listendefinition>* erlaubt es, die verfügbaren Listenparameter zu setzen. Dies geschieht, da es sich um Längen handelt, über die Befehle `\setlength` oder `\addtolength`. Wie sich eine Liste aufbaut, kann man in Abbildung 4.1 (Seite 75) sehen. Für nicht definierte Listenparameter werden Standardwerte angenommen.

Als Listenparameter stehen zur Verfügung:

`\topsep`

Vertikaler Zwischenraum, der zusätzlich zu `\parskip` zwischen den vorangehenden Text und die Liste bzw. die Liste und den nachfolgenden Text eingefügt wird.

`\partopsep`

Abstand, der zusätzlich zu `\topsep` und `\parskip` vor und nach der Liste eingefügt wird, wenn zwischen dem vorangehenden Text und dem ersten `\item` bzw. nach dem letzten `\item` und dem nachfolgenden Text eine Leerzeile auftritt.

`\parsep`

Abstand zwischen den Absätzen eines `\item`. Entspricht der Länge `\parskip` im normalen Text.

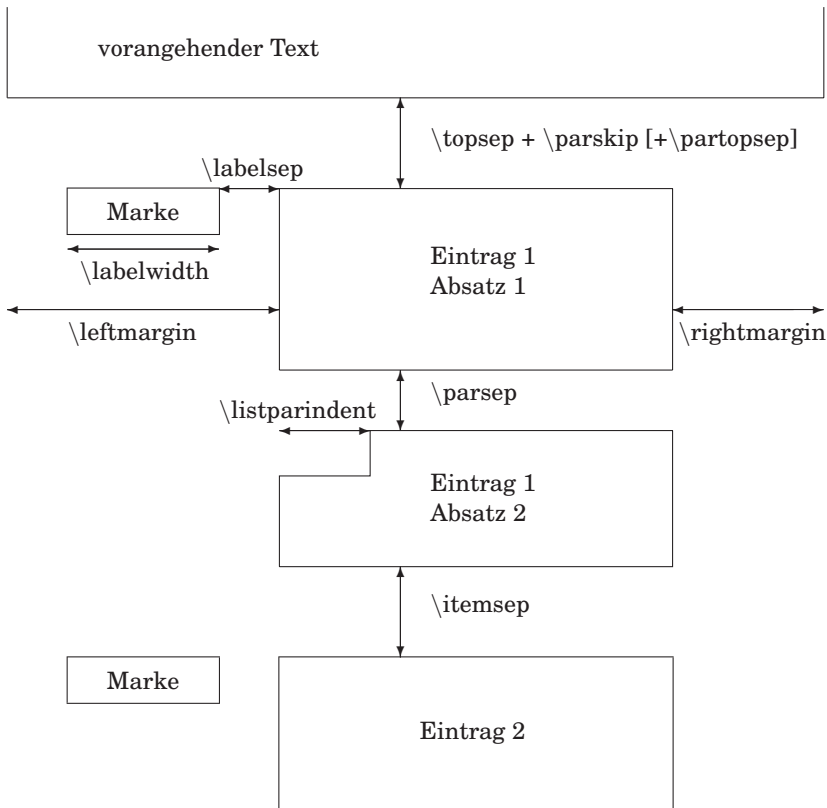


Abbildung 4.1: Aufbau einer Liste

`\itemsep`

Abstand, der zusätzlich zu `\parsep` zwischen zwei aufeinanderfolgende `\item` eingefügt wird.

`\leftmargin`

Linke Einrückungstiefe gegenüber dem linken Rand der augenblicklichen Umgebung.

`\rightmargin`

Rechte Einrückungstiefe gegenüber dem rechten Rand der augenblicklichen Umgebung.

`\listparindent`

Einrückungstiefe der ersten Zeile eines Absatzes innerhalb eines `\item` gegenüber dem linken Rand des Texts der Liste. Entspricht der Länge `\parindent` im normalen Text.

`\labelwidth`

Breite des Markierungsfeldes der Standardmarke. Die Markierung erscheint in diesem Feld rechtsbündig.

`\labelsep`

Abstand zwischen der Markierung und dem Listentext.

Um die Einträge einer Liste automatisch zu nummerieren, muss die Listendefinition die Anweisung `\usecounter{<Zähler>}` enthalten. Wobei `<Zähler>` der Name eines Zählers ist, der zuvor (außerhalb der Umgebung) mit `\newcounter{<Zähler>}` festgelegt wurde. Der Zähler wird zu Beginn der Liste auf 0 gesetzt und vor jedem `\item` ohne optionalem Argument um 1 erhöht.

Im nachfolgenden Beispiel wird für die Liste der zuvor definierte Zähler (Nummer) als Teil der Markierung in Form von kleinen römischen Buchstaben verwendet.

```

8 \newcounter{Nummer}
9 \setcounter{Nummer}{5} % Liste faengt bei 5 an
10 \begin{list}{\roman{Nummer}.}{\usecounter{Nummer}}
11 \setcounter{Nummer}{5}
12 \setlength{\leftmargin}{3em}
13 \setlength{\labelwidth}{4em}
14 \setlength{\labelsep}{.5em}
15 \setlength{\rightmargin}{\leftmargin}}
16 \item Der erste Listeneintrag.
17 Die Nummerierung beginnt bei sechs.
18 \item Der zweite Listeneintrag.
```

- vi. Der erste Listeneintrag. Die Nummerierung beginnt bei sechs.
- vii. Der zweite Listeneintrag.
- viii. Der dritte Listeneintrag.

[Buch1/SpezTextStruk/listen.tex]

Im Beispiel wird über `\setlength{\rightmargin}{\leftmargin}` das Setzen der Länge `\rightmargin` durch Zuweisen des aktuellen Werts von `\leftmargin` vorgenommen. Dies ist eine wichtige Möglichkeit, um Längen mit einem Wert zu belegen, der schon definiert wurde.

4.1.3.2 Die `trivlist`-Umgebung

Die `trivlist`-Umgebung ist eine sehr eingeschränkte Variante der `list`-Umgebung. Sie hat im Gegensatz zu `list` keine Argumente und verhält sich ansonsten wie eine `list`-Umgebung, bei der die Werte von `\leftmargin`, `\rightmargin`, `\labelwidth` und `\itemindent` auf Null (0 pt) gesetzt sind. Jedes `\item` in der Liste muss ein leeres optionales Argument haben. Im Normalfall findet diese Umgebung nur beim Erstellen von Makros Verwendung.

```

8 \begin{trivlist}
9 \centering\itshape
10 \item[] So kann man eine
11 Umgebung zentriert und mit
12 kursivem Text definieren.
13 \end{trivlist}
14 \end{document}

```

*So kann man eine Umgebung
zentriert und mit kursivem Text
definieren.*

[Buch1/SpezTextStruk/trivlist.tex]

Die `trivlist` liefert neben dem vertikalen Abstand auch eine Gruppierung, so dass Parameteränderungen am Ende der Liste wieder zurückgesetzt sind.

4.2 Tabellen

In Sachbüchern und anderen Dokumenten zu Sachthemen spielen Tabellen eine sehr wichtige Rolle, da sie Informationen in übersichtlicher und kurzer Form zusammenfassen. Tabellen sind üblicherweise aus Zeilen und Spalten aufgebaut. Meist sind sie wegen ihrer Höhe vom Fließtext abgesetzt.

Abgesetzte Tabellen werden – je nach Layout – durch Unter- oder Überschriften ergänzt und nummeriert. $\text{\LaTeX}$ stellt für diesen Zweck die Umgebung `table` zur Verfügung. Sie wird im Deutschen ebenfalls gern als *Tabelle* bezeichnet. Die Tabellenüber- oder -unterschrift wird dabei mit dem Wort *Tabelle* gefolgt von einer Nummer eingeleitet. In älteren Büchern wird dafür übrigens das Wort *Tafel* verwendet, das der englischen Bezeichnung *table* dieses Gleitobjekts entspricht, beschrieben in Abschn. 4.5 auf S. 102.

In diesem Abschnitt werden nur aus Zeilen und Spalten aufgebaute Tabellen behandelt, also eigentlich tabellarisches Material. Es existieren in $\text{\LaTeX}$ verschiedene Varianten, um Tabellen zu formatieren. Da ist zum

einen die `tabbing`-Umgebung. Hier wird eine Schreibmaschine mit Tabulatoren nachempfunden. Bei der `tabular`-Umgebung, der anderen Variante, wird auf der Basis der Spalteneinträge die Vermaßung der Tabelle bestimmt. Beide Umgebungen haben ihre Vor- und Nachteile. Daher wurden weitere Makropakete entwickelt, die das Erstellen von Tabellen weiter verbessern, unter anderem die Umgebungen `longtable` (s. 9.13, S. 266) und `supertabular` (nicht in diesem Band behandelt).

4.2.1 `tabbing`-Umgebung

Die `tabbing`-Umgebung kann man am Besten verstehen, wenn man sich eine Schreibmaschine mit Tabulatoren vorstellt. Ein Text wird tabellarisch dargestellt, indem man den Wagen immer auf die Tabulatoren vorrückt. Hier geschieht es ähnlich. Die Tabulatoren werden mit dem Befehl `\=` festgelegt. Sind sie einmal definiert, kann mit dem Befehl `\>` zur nächsten Spalte gerückt werden. Mit `\\` wird eine neue Zeile begonnen. Nach `\\` kann ein optionales Argument hinzugefügt werden, der den Abstand zur nächsten Zeile festlegt.

```

8 \begin{tabbing}
9   Regelm"a"sige K"orper \= Ecken \= Fl"achen \= Kanten
10                                     \\[3mm]
11   Tetraeder                \> vier   \> vier    \> sechs   \\
12   Oktaeder                \> sechs  \> acht    \> zw"olf   \\
13   Ikosaeder               \> zw"olf \> zwanzig \> drei"sig \\
14   Hexaeder                \> acht   \> sechs   \> zw"olf   \\
15 \end{tabbing}

```

Regelmäßige Körper Ecken Flächen Kanten				
Tetraeder	vier	vier	sechs	
Oktaeder	sechs	acht	zwölf	
Ikosaeder	zwölf	zwanzig	dreißig	
Hexaeder	acht	sechs	zwölf	

[Buch1/SpezTextStruk/tabb1.tex]

Bei diesem Beispiel ist vor den Worten ‚Ecken‘, ‚Flächen‘ und ‚Kanten‘ ein Tabulatorstopp positioniert. Eine neue Zeile wird nach einem größeren Abstand begonnen. Nach dem Setzen des Worts ‚Tetraeder‘ wird zum ersten Tabulatorstopp gerückt, dann die ‚vier‘ gesetzt, zum nächsten Tabulatorstopp gerückt, usw.

Da die Wörter ‚Regelmäßige Körper‘, ‚Ecken‘, ‚Flächen‘ und ‚Kanten‘ jeweils die längsten Wörter in den Spalten sind, funktioniert die Positionierung. Ersetzt man jedoch den Begriff ‚Regelmäßige Körper‘ durch ‚Körper‘

dann zeigt es sich, dass $\text{\LaTeX}$ im Gegensatz zu einer Schreibmaschine auch rückwärts positionieren kann.

```

8 \begin{tabbing}
9 K"orper      \= Ecken  \= Fl"achen \= Kanten \>[3mm]
10 Tetraeder    \> vier   \> vier    \> sechs   \[
11 Oktaeder     \> sechs  \> acht     \> zw"olf   \[
12 Ikosaeder    \> zw"olf \> zwanzig  \> drei"sig  \[
13 Hexaeder     \> acht   \> sechs    \> zw"olf   \[
14 \end{tabbing}

```

Körper	Ecken	Flächen	Kanten
Tetraeder	vier	vier	sechs
Oktaeder	sechs	acht	zwölf
Ikosaeder	zwölf	zwanzig	dreißig
Hexaeder	acht	sechs	zwölf

[Buch1/SpezTextStruk/tabb2.tex]

Bei $\text{\LaTeX}$ wird immer zum nächsten Tabulatorstopp gesprungen, auch wenn dies bedeutet, dass Text überdruckt wird.

Dieses Problem kann man umgehen, indem man eine Musterzeile zur Definition der Tabulatorstopps verwendet. Am Einfachsten trägt man in diese Musterzeile jeweils den längsten Begriff jeder Spalte ein. Mit dem Befehl `\kill` wird vermieden, die Musterzeile in die Ausgabe zu übernehmen. Wenn man etwas mehr Abstand zwischen den Spalten setzen möchte, so kann man in der Musterzeile zusätzliche Buchstaben einfügen, die dann im eigentlichen Text nicht sichtbar sind.

```

8 \begin{tabbing}
9 Ikosaederx    \= Eckenx \= zwanzigx \= Kantenx\kill
10 K"orper      \> Ecken  \> Fl"achen \> Kanten \>[3mm]
11 Tetraeder    \> vier   \> vier    \> sechs   \[
12 Oktaeder     \> sechs  \> acht     \> zw"olf   \[
13 Ikosaeder    \> zw"olf \> zwanzig  \> drei"sig  \[
14 Hexaeder     \> acht   \> sechs    \> zw"olf   \[
15 \end{tabbing}

```

Körper	Ecken	Flächen	Kanten
Tetraeder	vier	vier	sechs
Oktaeder	sechs	acht	zwölf
Ikosaeder	zwölf	zwanzig	dreißig
Hexaeder	acht	sechs	zwölf

Manchmal kann es erwünscht sein, die Tabulatorstopps neu zu positionieren. Mit dem Befehl `\pushtabs` können die momentan aktiven Positionen gespeichert werden. Danach können neue Tabulatoren gesetzt werden. Mit dem Befehl `\poptabs` können die gespeicherten Positionen wieder restauriert werden.

```

8 \begin{tabbing}
9 vierundzwanzig \= f"unfundzwanzig \= sechsundzwanzig \kill
10 einundzwanzig \> zweiundzwanzig \> dreiundzwanzig \\\
11 vierundzwanzig \> f"unfundzwanzig \> sechsundzwanzig \\\
12 \pushtabs
13 vier \= f"unf \= sechs \kill
14 eins \> zwei \> drei \\\
15 vier \> f"unf \> sechs \\\
16 \poptabs
17 einundddrei"sig \> zweiundddrei"sig \> dreiundddrei"sig \\\
18 dreiundddrei"sig \> f"unfundddrei"sig \> dreiundddrei"sig \\\
19 \end{tabbing}

```

```

einundzwanzig zweiundzwanzig dreiundzwanzig
vierundzwanzig fünfundzwanzig sechsundzwanzig
eins zwei drei
vier fünf sechs
einunddreißig zweiunddreißig dreiunddreißig
dreiunddreißig fünfunddreißig dreiunddreißig

```

Die Befehle `\pushtabs`/`\poptabs` können geschachtelt werden. Es muss jedoch auf jedes `\pushtab` ein `\poptab` folgen.

Der linke Textrand kann innerhalb einer `tabbing`-Umgebung modifiziert werden. Hierzu werden die Befehle `\+` und `\-` verwendet. Die Wirkungsweise kann man sich leicht erklären, wenn man die Tabulatoren durchnummeriert. Der linke Rand bekommt die 0, der erste Tabstopp die 1, usw. Mit jedem `\+` wird der linke Rand um 1 erhöht, d. h. der erste Tabstopp wird zum linken Rand. Ein erneuter Aufruf von `\+` macht den zweiten Stopp zum linken Rand. `\-` verringert die Zählung wieder um 1, d. h. der linke Rand wird auf den ersten Stopp positioniert.

```

8 \begin{tabbing}
9 siebzehn \= vierzehn \= neunzehn \= sechzehn \kill
10 eins \> zwei \> drei \> vier \+ \\\
11 sechs \> sieben \> acht \+ \\\
12 elf \> zw"olf \- \\\

```

```

13          vierzehn \> f"unfzehn \> sechzehn \-\
14 siebzehn \> achtzehn \> neunzehn \> zwanzig
15 \end{tabbing}

```

eins	zwei	drei	vier
	sechs	sieben	acht
		elf	zwölf
	vierzehn	fünfzehn	sechzehn
siebzehn	achtzehn	neunzehn	zwanzig

[Buch1/SpezTextStruk/tabb5.tex]

Man kann auch in jeder Zeile neue Tabulatoren mit \= einfügen.

Mit dem Befehl \' (Apostroph) wird der links vom \' stehende Text mit dem in \tabbingsep gespeicherten Abstand vor den nächsten Tabulatorstopp geschrieben. Dieser Abstand kann beispielsweise mit

```
\setlength{\tabbingsep}{10mm}
```

auf 10 mm gesetzt werden.

Der Befehl \` (Gravis) setzt den nachfolgenden Text rechtsbündig zum Seitenrand. Nach diesem Befehl darf kein \= und kein \> mehr folgen.

```

8 \begin{tabbing}
9 siebzehn \= vierzehn \= neunzehn \= sechzehn \kill
10 eins \> zwei \> drei \> vier \
11 \> neun \' zehn \>\> elf \' zw"olf \
12 \> sechzehn \
13 siebzehn \> achtzehn \> neunzehn \> zwanzig
14 \end{tabbing}

```

eins	zwei	drei	vier		
					acht
neun	zehn		elf	zwölf	
					sechzehn
siebzehn	achtzehn	neunzehn	zwanzig		

[Buch1/SpezTextStruk/tabb6.tex]

Der Befehl \< springt einen Tabulatorstopp nach links. Dies macht natürlich nur Sinn, wenn mit \+ der linke Rand versetzt wurde und temporär nach links gesprungen werden soll.

Normalerweise sind die Befehle `\=`, `\`` und `\'` dazu gedacht, Akzente zu setzen. Sie werden jedoch innerhalb der `tabbing`-Umgebung redefiniert. Die Befehle `\a=`, `\a`` und `\a'` erzeugen bei Bedarf die entsprechenden Akzente.

```

8 \begin{tabbing}
9 xxxxxxxxxxxxxxxx \= xxxxxxxxxxxxxxxx \kill
10 \a`a droite \> \a'a gauche \\
11 \a'a c\^ot\`a'e de \> en face de \\
12 \end{tabbing}

```

	à droite	à gauche
	à côté de	en face de

[Buch1/SpeczTextStruk/tabb7.tex]

Es ist nicht möglich, die `tabbing`-Umgebung zu schachteln.

4.2.2 tabular-Umgebung

In diesem Abschnitt werden aus Zeilen und Spalten aufgebaute Tabellen behandelt, die im Textmodus mit Hilfe der Umgebung `tabular` gesetzt werden. Für entsprechend strukturierte Objekte im mathematischen Modus – etwa Elemente von Matrizen und Determinanten – stellt $\text{\LaTeX}$ die gleichartige Umgebung `array` zur Verfügung (vgl. 6.2.5, S. 152). $\text{\LaTeX}$ baut die Umgebung `tabular` übrigens auf dem `array` auf.

Seitenumbrüche sind in der `tabular`-Umgebung nicht zugelassen. Zum Satz von Tabellen, die sich über mehrere Seiten erstrecken können, stehen die Umgebungen `longtable` (s. 9.13, S. 266) und `supertabular` (in einem späteren Band behandelt) zur Verfügung.

Die Standard-Umgebung zum Setzen einer Tabelle hat die folgende Form:

```

\begin{tabular}[<VPos>]{<Vereinbarungen>
  <Zeile> [\\[<Abstand>]]
  <Zeile> ] ...
\end{tabular}

```

$\text{\LaTeX}$ baut die Tabelle auf und setzt sie anschließend wie ein einzelnes Zeichen – etwa einen Buchstaben oder ein Wort – an der aktuellen Stelle in den Text. Die Breite der Tabelle bestimmt $\text{\LaTeX}$ aus der Breite der Spalten. Abgesehen vom Anpassen der Zeilenabstände an den Platzbedarf, wird der Fließtext durch eine Tabelle nicht unterbrochen.

Durch das optionale Argument `<VPos>` lässt sich die vertikale Ausrichtung der Tabelle gegenüber der Grundlinie festlegen. Fehlt das optionale Argument, wird sie gegenüber der Grundlinie zentriert. Durch Angabe von `t` lässt sich die Tabelle so ausrichten, dass die oberste Zeile der Tabelle auf der Grundlinie steht. Die Angabe `b` setzt die unterste Tabellenzeile auf die Grundlinie.

Der Aufbau der Tabelle wird durch das Argument `<Vereinbarungen>` beschrieben. Für jede Spalte der Tabelle legt dieses Argument fest, wie Einträge in der Spalte auszurichten oder umzuberechnen sind. Auch die Übergänge von einer Spalte zur nächsten oder linke und rechte äußere Begrenzungen der ersten und letzten Spalte – beispielsweise ein senkrechter Strich zwischen zwei Spalten oder am Tabellenrand – werden an dieser Stelle festgelegt. Die Angaben legen die Spalten und Spaltenübergänge bzw. Begrenzungen in der Reihenfolge ihres Auftretens von links nach rechts fest.

Durch die Angabe einer Länge `<Abstand>` ist es möglich, nach einer Zeile den Zeilenabstand um den angegebenen Wert vergrößern bzw. im Fall einer negativen Länge zu verringern.

L^AT_EX kann auch Tabellen mit vorgegebener Breite setzen. Für diesen Zweck steht eine Variante der `tabular`-Umgebung zur Verfügung, bei der die Breite als zusätzliches Argument `<Breite>` angegeben wird:

```
\begin{tabular*}{<Breite>}[<VPos>]{<Vereinbarungen>}
  <Zeile> [\\[<Abstand>]]
  <Zeile> ] ...
\end{tabular*}
```

Die Anpassung der Breite erfolgt durch Veränderung der Abstände zwischen den Spalten. Dazu sind Spaltentrennungen, bei denen der Abstand angepasst werden kann, in der Form `@{<Trennung>}` anzugeben. `<Trennung>` muss dabei einen Befehl `\extracolsep{<Gummilänge>}` enthalten.

Die Breite von Spalten, deren Ausrichtung vorgegeben ist, wird durch den breitesten Eintrag in der Spalte festgelegt. Ein Zeilenumbruch wird nicht vorgenommen, selbst wenn dadurch die Tabelle breiter wird als die Textbreite eigentlich zulässt. Alternativ kann eine Spalte mit einer vorgegebenen Breite vereinbart werden, deren Einträge im Blocksatz gesetzt und bei Bedarf umbrochen werden.

Vorgesehen sind für Spalten und die Übergänge zwischen Spalten – dazu gehören auch die linke Begrenzung der ersten und die rechte Begrenzung der letzten Spalte – die folgenden Angaben:

Spaltenvereinbarungen

- c** Eine Spalte, deren Einträge zentriert gesetzt werden:
Die Einträge werden nicht umbrochen, der breiteste Eintrag bestimmt die Spaltenbreite.
- l** Eine Spalte mit linksbündig gesetzten Einträgen:
Die Einträge werden nicht umbrochen, der breiteste Eintrag bestimmt die Spaltenbreite.
- r** Eine Spalte, deren Einträge rechtsbündig gesetzt werden:
Die Einträge werden nicht umbrochen, der breiteste Eintrag bestimmt die Spaltenbreite.

p{<Länge>}

Eine Spalte der Breite <Länge>, deren Einträge im Blocksatz gesetzt werden. Sie werden umbrochen, wenn sie mehr Platz benötigen als die zur Verfügung stehende Breite. Kürzere Zeilen und die letzte Zeile bei mehrzeiligen Einträgen werden linksbündig gesetzt.

*Spaltentrennungen***keine Angabe**

Ist keine Spaltentrennung angegeben, werden zwei Spalten durch den zweifachen Standardabstand `\tabcolsep` getrennt.

- |** Ein (einzelter) senkrechter Strich zwischen zwei Spalten in der Mitte des (unveränderten) normalen Spaltenabstands.
Durch Wiederholung von `|` können zusätzliche senkrechte Striche eingefügt werden, die dann allerdings den Abstand der Spalten jeweils um den vorgegebenen Abstand `\doublerulesep` von Strichen erhöhen.

@{<Trennung>}

Die Spalten werden durch <Trennung> getrennt. Die sonst vorgesehenen Abstände entfallen; gegebenenfalls lassen sich durchgezogene senkrechte Striche in <Trennung> (wie bei der Spaltentrennung `|`) mit dem Befehl `\vline` einfügen. Der Spaltenabstand wird durch <Trennung> ersetzt.

*Wiederholung von Teilvereinbarungen****{<Anzahl>}{<Teilvereinbarungen>}**

Diese Angabe dient zum Abkürzen sich wiederholender Angaben in den <Vereinbarungen> für die gesamte Tabelle. Die Wirkung ist die gleiche wie <Anzahl> direkt aufeinanderfolgender Angaben der <Teilvereinbarungen> in den <Vereinbarungen>. Die Angabe `*{3}{c|}` hat also die gleiche Wirkung wie `c|c|c|`.

Die Einträge in die Tabellen-Spalten werden ebenso wie das Argument *<Trennung>* von @ im Textmodus gesetzt.

Das folgende Beispiel demonstriert die Varianten der Spaltendefinitionen:

```

8 \begin{tabular}{l*{2}{c@{~--~}}r|p{3cm}||}
9 \textbf{Links} & \bfseries Zentriert & Zentriert
10 & \textbf{Rechts} & \textbf{Zeilenumbruch}\\[3mm]
11 links- & in der & ebenfalls & rechts-
12 & \multicolumn{3}{l}{mehrzeilig mit Zeilenumbruch\\}
13 b"undig & Mitte & zentriert & b"undig & mit explizitem\\
14 & & & & Zeilenumbruch\\
15 & & zentriert
16 \end{tabular}

```

Links	Zentriert	–	Zentriert	–	Rechts	Zeilenumbruch
links-	in der	–	ebenfalls	–	rechts-	mehrzeilig mit Zeilenumbruch
bündig	Mitte	–	zentriert	–	bündig	mit explizitem Zeilenumbruch
	zentriert	–				

[Buch1/SpezTextStruk/tabular1.tex]

Das Beispiel zeigt in Zeile 8 die beschriebenen Spaltenvereinbarungen und Spaltentrennungen. Die Spaltenvereinbarung für die zentrierte Spalte wird zusammen mit der ihr folgenden Spaltentrennung wiederholt. Die Einträge für die zweite und dritte Spalte in Zeile 9 zeigen, dass eine (globale) Schriftumschaltung wie `\bfseries` auf die Spalte beschränkt ist, in der sie verwendet wird.

In Zeile 14 bleiben die ersten vier Spalten leer. In der letzten Tabellenzeile 15 ist zu sehen, dass bei einem vorzeitigen Zeilenende der Rest der Zeile leer bleibt, d. h. weitere Spaltentrennungen nicht mehr gesetzt werden. Sollen die fehlenden Spaltentrennungen gesetzt werden, müssen wie in Zeile 14 leere Spalten angegeben werden.

Die Zeilen 12–14 zeigen, dass führende und angehängte Leerzeichen bei den Spalteneinträgen keine Rolle spielen.

Mit Hilfe der Vereinbarungen am Tabellenanfang lassen sich nur einfache Tabellen aufbauen, die keine spaltenübergreifenden Einträge oder horizontale Linien enthalten. Für solche Ergänzungen stellen die Umgebungen `tabular` und `tabular*` spezielle Befehle zur Verfügung, die nur in diesen Umgebungen definiert sind.

```
\multicolumn{<Spzahl>}{<Vereinbarungen>}{<Eintrag>}
```

- Spaltenübergreifende Einträge lassen sich mit einem `\multicolumn`-Befehl setzen. Das erste Argument `<Spzahl>` legt die Anzahl der Spalten einschließlich der aktuellen Spalte fest, über die sich der damit gesetzte Eintrag erstreckt. `<Spzahl>` darf die Anzahl der verbleibenden Spalten – einschließlich der Spalte, in der der Befehl verwendet wird – nicht übersteigen.
- Das zweite Argument legt fest, wie die Spalte zu setzen und rechts zu begrenzen ist. Bei Verwendung in der ersten Spalte ist auch die linke Spaltenbegrenzung anzugeben, die andernfalls bereits gesetzt ist. Da mit `\multicolumn` genau ein Eintrag gesetzt wird, muss auch genau eine Spaltenvereinbarung (`c`, `l`, `r` oder `p{...}`) angegeben sein. Spaltentrennungen können hingegen mehrfach angegeben werden.
- Das letzte Argument `<Eintrag>` ist der spaltenübergreifende Eintrag, der in der Form gesetzt wird, die in `<Vereinbarungen>` für den Eintrag festgelegt wurde.
- Mit `<Spzahl> = 1` kann der Befehl `\multicolumn` auch verwendet werden, um einen Eintrag in der aktuellen Spalte abweichend von der für diese Spalte vorgesehenen Form zu setzen oder die der Spalte folgende Spaltentrennung abzuändern. Eine häufige Anwendung dieser speziellen Form sind Spaltenüberschriften, die abweichend von den Inhalten der Spalten ausgerichtet werden. In den übrigen Zeilen wird diese Form eher selten eingesetzt.

Daneben gibt es einige weitere Befehle, die in Tabellen für Spezialaufgaben eingesetzt werden können.

```
\vline
```

Entspricht einer Spaltentrennung `|`, kann aber im Unterschied dazu auch im Argument von `@{...}` und sogar in einem Eintrag verwendet werden. Die Höhe der vertikalen Linie passt sich an die Höhe der Zeile an.

```
\hline
```

Mit diesem Befehl wird zwischen zwei Tabellenzeilen eine horizontale Linie über die gesamte Breite der Tabelle gesetzt. Der Befehl kann also nur vor der ersten Zeile, nach einem Zeilenende `\\` oder nach einem vorhergehenden `\hline`-Befehl verwendet werden.

Mehrere `\hline`-Befehle zwischen zwei Zeilen einer Tabelle setzen die entsprechende Anzahl horizontaler Linien zwischen den

Zeilen. Der Abstand zwischen den Linien wird durch die im Folgenden beschriebene Länge `\doublerulesep` festgelegt.

```
\cline{<Startspalte>-<Endspalte>}
```

Dieser Befehl erlaubt es, horizontale Linien über vorgegebene Spalten zwischen zwei Tabellenzeilen zu setzen. Er kann an jeder Stelle verwendet werden, an der ein `\hline`-Befehl stehen kann. Die horizontale Linie beginnt in Spalte `<Startspalte>` und endet in Spalte `<Endspalte>`. Beide Angaben sind erforderlich. Um eine Linie in nur einer Spalte zu setzen, ist die entsprechende Spalte sowohl als Start- als auch als Endspalte anzugeben.

Wie der `\hline`-Befehl kann auch `\cline` zwischen zwei Zeilen mehrfach verwendet werden, beispielsweise im Fall unterbrochener horizontaler Linien. Aus diesem Grund werden alle mit `\cline` gesetzten horizontalen Linien auf der gleichen Höhe gesetzt. Für Mehrfachlinien müssen daher Leerzeilen eingefügt werden, bei denen dann aber in der Regel der Zeilenabstand korrigiert werden muss.

```
\extracolsep{<Länge>}
```

Die Verwendung dieses Befehls in einem `@`-Ausdruck fügt nach den linken Spaltentrennungen *aller folgenden Spalten* den als Länge angegebenen Abstand ein. Die genaue Position von `\extracolsep` im `@`-Ausdruck spielt dabei keine Rolle. Ein folgender `@`-Ausdruck unterdrückt diesen zusätzlichen Abstand nicht. Eine erneute Verwendung überschreibt den bisherigen Wert. Pro Spaltentrennung mit `@` darf `\extracolsep` allerdings *höchstens einmal* verwendet werden.

`\extracolsep{0pt}` stellt für alle folgenden Spalten den normalen Abstand wieder her. Dabei ist jedoch zu beachten, dass ein `@`-Ausdruck ohne Abstand zu den links und rechts von ihm stehenden Spalten gesetzt wird. Gegebenenfalls muss der fehlende Abstand explizit eingefügt werden. Eine einfachere Lösung bietet das Paket `array` (siehe 9.2), das eine Alternative zu `@`-Ausdrücken bietet, bei der die normalen Abstände erhalten bleiben.

Unentbehrlich ist der Befehl `\extracolsep` bei der Umgebung `tabular*`, da die Tabelle durch Einfügen von Abständen zwischen den Spalten auf die gewünschte Breite gebracht werden muss. In diesem Fall ist zumindest in einem Fall als Argument von `\extracolsep` eine Gummilänge – etwa `\fill` oder `5cm minus 5cm` – anzugeben. Bei mehrfacher Verwendung mit Gummilängen wird der Platz anteilmäßig auf die einzelnen Gummilängen verteilt (`40mm minus 36mm` bewirkt den doppelten Abstand wie `20mm minus 18mm`, wenn beide vorkommen).

Außerdem ist bei Verwendung von `\cline`-Befehlen zu beachten, dass die horizontale Linie im Bereich eines `\extracolsep` nur gezogen wird, wenn `\extracolsep` innerhalb des angegebenen Spaltenbereichs liegt. Vor der Startspalte und nach der Endspalte wird die Linie in `\extracolsep`-Bereichen nicht gezeichnet.

`\tabularnewline[<Zeilenabstand>]`

In Tabellen eine alternative Bezeichnung für `\` und außerhalb ein Leerbefehl (`\relax`).

```
8 \begin{tabular}{|l *{2}{@{~--~}c} | r | p{3cm} | }
9 \hline
10 \multicolumn{1}{|c|}{\textbf{Links}\rule{2.5ex}{0pt}}
11 & \multicolumn{1}{c}{\textbf{Zentriert}}
12 & \multicolumn{1}{c}{\textbf{Zentriert}}
13 & \multicolumn{1}{c}{\textbf{Rechts}}
14 & \multicolumn{1}{c}{\textbf{Zeilenumbruch}}\relax[1mm]
15 \hline
16 links-&\multicolumn{2}{c@{~--~}}{zweispaltig zentriert}
17 & rechts- & mehrzeilig mit Zeilenumbruch\relax
18 b"undig & Mitte & zentriert & b"undig & mit explizitem\relax
19 \multicolumn{3}{|c|}{&& Zeilenumbruch\relax}
20 \cline{1-1}\cline{2-3}\cline{5-5}
21 \cline{1-2}
22 & \multicolumn{1}{c@{~\vline~}c@{~\vline~}}{zen\vline
23 triert} & \multicolumn{1}{c|}{&&\relax}
24 \hline
25 \end{tabular}
26 \end{document}
```

Links	Zentriert	Zentriert	Rechts	Zeilenumbruch		
links-	–	zweispaltig zentriert	–	rechts-	mehrzeilig mit Zeilenumbruch	
bündig	–	Mitte	–	zentriert	bündig	mit explizitem Zeilenumbruch
	–	zentriert				

[Buch1/SpezTextStruk/tabular2.tex]

Die `\hline`-Befehle in den Zeilen 9 und 24 bilden zusammen mit den in Zeile 8 vereinbarten vertikalen Linien vor der ersten und nach der letzten Spalte einen Rahmen um die Tabelle. Nach den Spaltenüberschriften in den Zeilen 10–14 folgt eine weitere horizontale Linie quer durch die gesamte Tabelle.

In den Zeilen 20 und 21 werden mit dem `\cline`-Befehl auf Spalten beschränkte horizontale Linien gesetzt. Der erste und dritte Befehl in Zeile 20 beschränkt die Linie auf eine einzelne Spalte, indem die Nummer der jeweiligen Spalte sowohl als Anfangs- wie auch als Endspalte angegeben wird. Ein vereinfachter Befehl ist für diesen Sonderfall nicht vorgesehen. Der mittlere Befehl in dieser Zeile setzt eine Linie über die zweite und dritte Spalte. Da der erste Befehl in der Zeile eine Horizontale in der ersten Spalte zeichnet, ergibt sich im Ergebnis eine zusammenhängende Linie über die ersten drei Spalten, da die beiden Teile der Linie auf der gleichen Höhe liegen. Diese Eigenschaft – dass alle mit `\cline` zwischen zwei Tabellenzeilen gesetzten Linien auf der gleichen Höhe gezeichnet werden – wird besonders deutlich durch den `\cline`-Befehl in Zeile 21, der erneut eine Linie über die beiden ersten Spalten zeichnet. Die Ausgabe zeigt, dass diese Linie mit den zuvor in diesen Spalten gezeichneten Linien überlagert wird und daher keine eigene Wirkung hat. Die Zeilen 20 und 21 könnten daher verkürzt werden zu `\cline{1-3}\cline{5-5}`.

Von großer Bedeutung beim Satz von Tabellen ist der `\multicolumn`-Befehl. Das Beispiel zeigt verschiedene Anwendungen des Befehls. Die Grundform zeigt Zeile 16, in der beginnend in der zweiten Spalte ein zweispaltiger Eintrag zentriert gesetzt wird. Die linke Spaltenbegrenzung wurde bereits beim Übergang von der ersten zur zweiten Spalte gesetzt. Die Vereinbarungen des `\multicolumn`-Befehls enthalten daher nur ein `c` zum Zentrieren des Eintrags und die rechte Spaltenbegrenzung.

Die `\multicolumn`-Befehle in den Zeilen 10–14 haben eine andere Funktion und dienen dazu, die Ausrichtung der Einträge und die rechte Spaltenbegrenzung lokal abzuändern. Aus diesem Grund sind diese Befehle auch auf eine einzelne Spalte – die aktuelle – beschränkt. Davon abgesehen gibt es lediglich die Besonderheit, dass beim `\multicolumn`-Befehl in Zeile 10 auch die linke Spaltenbegrenzung gesetzt wird. Das ist notwendig, da in der ersten Spalte eine linke Spaltenbegrenzung (noch) nicht gesetzt wurde. Im Argument dieses `\multicolumn`-Befehls ist übrigens mit `\rule` ein unsichtbarer vertikaler Strich eingefügt, um den Abstand zur horizontalen Linie über der Zeile zu vergrößern.

Wieder eine andere Funktion hat der `\multicolumn`-Befehl in Zeile 19. Er entfernt in der Tabellenzeile die Spaltenbegrenzungen zwischen den ersten vier Spalten, auch wenn er sich nur über die ersten drei Spalten erstreckt. Das macht erneut deutlich, dass die Spaltenbegrenzung am Ende eines `\multicolumn`-Eintrags nicht aus der Tabellenvereinbarung übernommen wird, sondern im `\multicolumn`-Befehl vereinbart werden muss. Der `\multicolumn`-Befehl in Zeile 23 ist hingegen entbehrlich,

da die (leere) Spalte ohnehin durch eine vertikale Linie rechts begrenzt ist.

Mit Hilfe des `\multicolumn`-Befehls in den Zeilen 22 und 23 wird die Wirkung des `\vline`-Befehls gezeigt. In den Argumenten von `@` wird er verwendet wie `|` außerhalb von Spaltentrennungen mit `@`; `|` kann in `@`-Ausdrücken nicht verwendet werden. Eine zweite Anwendung von `\vline` wird am Ende von Zeile 22 deutlich, wo er innerhalb eines Spalteneintrags verwendet wird, um mitten im Eintrag eine senkrechte Linie wie bei einer Spaltentrennung zu setzen.

```
8 \begin{tabular*}{4cm}{|cc@{\extracolsep{\fill}}
9 c@{\vline}c@{\hspace{12pt}}\extracolsep{0pt}}c|}
10 \hline
11 a & b & & c & d\\ \cline{1-2}\cline{3-3}\cline{4-5}
12 u & v & & w & x\\w & x & & y & z\\
13 \hline
14 \end{tabular*}
15 \hspace{1cm}
16 \begin{tabular}{|cc@{\extracolsep{4mm}}\hspace{12pt}}cc|}
17 \hline
18 a & b & c & d\\ \cline{1-2}\cline{3-4}
19 u & v & w & x\\w & x & y & z\\
20 \hline
21 \end{tabular}
```

a	b	c	d	a	b	c	d
u	v	w	x	u	v	w	x
w	x	y	z	w	x	y	z

[Buch1/SpezTextStruk/tabular3.tex]

Dieses zweite Beispiel zeigt die Variante `tabular*` der Tabellen und die Verwendung des Befehls `\extracolsep` in beiden Tabellenumgebungen.

Bei der Umgebung `tabular*` wird in Zeile 8 als erstes Argument die Breite der zu setzenden Tabelle – hier 4 cm – angegeben. Die Festlegung der Spalten und Spaltentrennungen in den Zeilen 8 und 9 zeigt einige Besonderheiten.

Obwohl nur vier Spalten in der Tabelle verwendet werden, sind in der Tabellenvereinbarung fünf Spalten angegeben. Die dritte Spalte – das `c` am Anfang von Zeile 9 – bleibt leer. Sie ist allerdings erforderlich, um die senkrechte Linie in der Mitte der Tabelle in der Mitte des freien Raumes zu zentrieren, da der Abstand zwischen der zweiten und vierten Spalte dehnbar sein soll. Dazu muss bei Tabellen mit vorgegebener Breite – wie

bereits erwähnt – mindestens eine dehnbare Spaltentrennung mit Hilfe eines `\extracolsep`-Befehls angegeben werden. Nun soll die dehnbare Spaltentrennung in der Mitte der Tabelle liegen, und andererseits soll in der Mitte eine vertikale Linie stehen. Mit einem einzigen @-Ausdruck ist das aber nicht möglich, da `\extracolsep` den Abstand stets *am Anfang der folgenden Spalten* setzt. Die leere dritte Spalte wird nun benötigt, um an ihrem Anfang den dehnbaren Abstand `\extracolsep{\fill}` links von der senkrechten Linie (`@{\vline}`) zu setzen, die die dritte Spalte rechts begrenzt. Am Anfang der vierten Spalte und damit nach der Linie wird der dehnbare Abstand automatisch erneut gesetzt.

Zwischen der vierten und der fünften Spalte soll wieder ein normaler Spaltenabstand freigehalten werden. Dazu wird der aktive (dehnbare) zusätzliche Spaltenabstand in Zeile 9 mit `\extracolsep{0pt}` zurückgesetzt. Da das im Argument eines @-Ausdrucks erfolgt, würden die vierte und fünfte Spalte bei fehlendem `\hspace{12pt}` ohne Abstand direkt aufeinander folgen. Die Reihenfolge von `\hspace` und `\extracolsep` spielt übrigens keine Rolle, wie auch die zweite Tabelle in diesem Beispiel zeigt.

Die drei `\cline`-Befehle in Zeile 11 über die Spalten 1 und 2, über Spalte 3 sowie die Spalten 4 und 5 zeigen das Verhalten bei angrenzenden Zusatzabständen mit `\extracolsep`. Wie bereits erwähnt, erstrecken sich die horizontalen Linien nicht über die links und rechts anschließenden Zusatzabstände zwischen den Spalten 2 und 3 sowie 3 und 4. Spalte 3 ist leer und hat daher keine Breite; deswegen hat `\cline{3-3}` auch keine sichtbare Wirkung. Enthielte die Tabelle keine Zusatzabstände, hätten die drei `\cline`-Befehle die gleiche Wirkung wie ein `\hline`-Befehl, da alle Spalten berücksichtigt wurden.

Die zweite Tabelle in den Zeilen 16–22 zeigen die Verwendung von `\extracolsep` in einem `tabular` zur Vergrößerung der Spaltenabstände um 4 mm. Bei der Festlegung des Zusatzabstands zwischen der zweiten und dritten Spalte wird zusätzlich ein Abstand mit `\hspace{12pt}` eingefügt, der dem üblichen Spaltenabstand entspricht. Wegen der Spaltentrennung mit einem @-Ausdruck würde dieser Abstand andernfalls fehlen. Obwohl `\hspace` *nach* `\extracolsep` angegeben ist, zeigen die `\cline`-Befehle in Zeile 19, dass `\extracolsep` erst *nach* der Spaltentrennung durch den Abstand `\hspace{4mm}` am Anfang der folgenden Spalte wirksam wird: `\cline{1-2}` erstreckt sich offensichtlich bis zum Ende der Spaltentrennung ohne den Zusatzabstand nach rechts. Auch hier würde die Befehlsfolge `\cline{1-2}\cline{3-4}` wie ein `\hline`-Befehl wirken, wenn zwischen den Spalten 2 und 3 kein Zusatzabstand eingefügt wäre.

Weitere Eigenschaften von Tabellen sind zum Layout zu rechnen. Dazu gehören beispielsweise die Dicke der Linien, der Linienabstand bei Mehrfachlinien oder der voreingestellte Spaltenabstand. Diese Abstände werden durch Längen festgelegt. Der Zeilenabstand lässt sich durch einen Befehl verändern, der als Faktor dem Zeilenabstand vorangestellt wird. Die Einstellungen gelten – mit Ausnahme des Spaltenabstands – sowohl für die hier behandelten Tabellen als auch für das Äquivalent im mathematischen Modus. Die folgenden Stilparameter legen die erwähnten Abstände für folgende Tabellen fest. Innerhalb einer Tabelle lassen sich die Stilparameter nicht mehr ändern.

`\tabcolsep`

Diese Länge legt den Spaltenabstand in Tabellen fest. Der Wert der Länge ist der *halbe* Spaltenabstand.

(Voreinstellung: 6pt)

Bei der entsprechenden Umgebung `array` im mathematischen Modus übernimmt die Länge `\arraycolsep` diese Aufgabe.

`\arrayrulewidth`

Die Strichstärke der horizontalen und vertikalen Linien wird durch diese Länge festgelegt. Dabei ist zu beachten, dass sie bei der Festlegung der Spaltenabstände so behandelt werden, als hätten sie keine eigene Strichstärke. Eine Verstärkung der Striche rückt die Zeilen und Spalten also nicht weiter auseinander.

(Voreinstellung: 0.4pt)

`\doublerulesep`

Zwei oder mehr parallele horizontale oder vertikale Linien müssen mit einem kleinen Abstand gesetzt werden, damit sie unterscheidbar sind und nicht zu einer dicken Linie verschmelzen.

`\doublerulesep` legt den Abstand der Mitte zweier paralleler Linien – unabhängig von ihrer Dicke – fest. Für dicke Linien muss der Abstand daher erhöht werden, damit sie unterscheidbar sind.

(Voreinstellung: 2pt)

`\arraystretch`

Hier handelt es sich um einen Befehl, der als Faktor beim Zeilenabstand berücksichtigt wird. Er muss einen Zahlenwert enthalten. Tabellenzeilen haben standardmäßig den gleichen Abstand `\baselineskip` wie gewöhnliche Textzeilen.

`\renewcommand{\arraystretch}{1.5}`

vergrößert den Zeilenabstand in Tabellen um 50 %,

`\renewcommand{\arraystretch}{.8}`

verringert ihn auf 80 % des normalen Zeilenabstands. Änderungen des Werts sind nur für nachfolgende Tabellenumgebungen

wirksam, da der Zeilenabstand bereits beim Beginn der Tabellenumgebung für die gesamte Tabelle festgelegt wird.
(Voreinstellung: 1)

```

8 \begin{tabular}{|c||c|}
9   \hline x&y\\\hline\hline g&h\\\hline
10 \end{tabular}\qqquad
11 {\setlength{\tabcolsep}{12pt}}
12 \begin{tabular}{|c||c|}
13   \hline x&y\\\hline\hline g&h\\\hline
14 \end{tabular}}\qqquad
15 {\setlength{\arrayrulewidth}{2pt}}
16 \begin{tabular}{|c||c|}
17   \hline x&y\\\hline\hline g&h\\\hline
18 \end{tabular}\qqquad
19 \setlength{\doublerulesep}{3pt}}
20 \begin{tabular}{|c||c|}
21   \hline x&y\\\hline\hline g&h\\\hline
22 \end{tabular}}\qqquad
23 {\renewcommand{\arraystretch}{1.5}}
24 \begin{tabular}{|c||c|}
25   \hline x&y\\\hline\hline g&h\\\hline
26 \end{tabular}}
27 \end{document}

```

The image shows five tables side-by-side, each with two rows and two columns. The first table (lines 8-10) has a standard appearance. The second table (lines 12-14) has a wider column gap. The third table (lines 16-18) has thicker vertical lines. The fourth table (lines 20-22) has a larger gap between the vertical lines. The fifth table (lines 24-27) has a larger row gap.

x	y
g	h

x	y
g	h

x	y
g	h

x	y
g	h

x	y
g	h

[Buch1/SpecTextStruk/tabular4.tex]

Das Beispiel zeigt die Wirkung der Stilparameter anhand einer einfachen Tabelle. Die Tabellen in den Zeilen 8–14 sind mit den Voreinstellungen gesetzt. Für die Tabelle in den Zeilen 12–14 wurde in Zeile 11 der Spaltenabstand gegenüber der Voreinstellung verdoppelt. In Zeile 15 wird für die beiden folgenden Tabellen die Liniendicke um 150 % vergrößert. Bei der Tabelle in den Zeilen 16–18 verschmelzen die beiden parallelen Linien, da der Abstand der Linien nicht erhöht wurde. Für die Tabelle in den Zeilen 20–22 wurde daher in Zeile 19 der Linienabstand soweit vergrößert, dass die beiden Linien unterscheidbar sind. In Zeile 23 wird der Zeilenabstand für die Tabelle in den Zeilen 24–27 um 50 % erhöht.

Im Beispiel zeigen sich auch einige Probleme mit horizontalen und vertikalen Linien in Tabellen. Einerseits ist erkennbar, dass die vertikalen Linien zwischen den beiden direkt aufeinander folgenden horizontalen Linien unterbrochen sind, während die horizontalen Linien nicht

unterbrochen sind. Abhilfe für dieses Problem bietet das Paket `hhline` (s. 9.10, S. 258), mit dem das Verhalten beim Kreuzen zweier Doppellinien oder einer Doppellinie mit einer einfachen Linie festgelegt werden kann. Mit größerem Aufwand ließe sich das Problem auch durch Einfügen einer Leerzeile mit einer geeigneten Verkürzung des Zeilenabstands lösen.

Auf der anderen Seite ist bei der dritten und vierten Tabelle mit den dicken Linien zu sehen, dass die horizontalen Linien vorzeitig mitten in den vertikalen Linien abbrechen, während die vertikalen Linien jeweils am Rand der horizontalen enden. Bei den von $\text{\LaTeX}$ vorgesehenen dünnen Linien ist dieser Defekt mit bloßem Auge nicht zu erkennen. Das Paket `array` (s. 9.2, S. 230) korrigiert diesen Fehler. Darüber hinaus stellt das Paket Erweiterungen zur Verfügung, die den Satz von Tabellen verbessern.

Zum Abschluss noch ein Beispiel, wie man Tabellen schachteln kann. Die dazu notwendige Umgebung `minipage` wird in Abschn. 5.3.2, S. 129 behandelt.

```

8 \setlength{\tabcolsep}{5pt}
9 \begin{document}
10 \begin{tabular}[t]{t}{|c|c|}
11 \hline

```

<table><tr><td>x</td><td></td><td></td></tr><tr><td></td><td></td><td>o</td></tr><tr><td></td><td>x</td><td></td></tr></table>	x					o		x		<table><tr><td>x</td><td></td><td></td></tr><tr><td></td><td></td><td>o</td></tr><tr><td>x</td><td></td><td>x</td></tr></table>	x					o	x		x
x																			
		o																	
	x																		
x																			
		o																	
x		x																	
<table><tr><td>x</td><td></td><td></td></tr><tr><td></td><td>x</td><td>o</td></tr><tr><td></td><td>o</td><td></td></tr></table>	x				x	o		o		<table><tr><td>x</td><td></td><td>x</td></tr><tr><td></td><td>o</td><td>o</td></tr><tr><td>x</td><td>x</td><td></td></tr></table>	x		x		o	o	x	x	
x																			
	x	o																	
	o																		
x		x																	
	o	o																	
x	x																		

[Buch1/SpezTextStruk/tabb8.tex]

4.3 Textverschiebungen

$\text{\LaTeX}$ setzt seine Textausgabe immer im Blocksatz. Hierunter versteht man, dass der gesetzte Text rechts und links einen gleichmäßigen Rand

hat. Das Gegenteil vom Blocksatz ist der Flattersatz. Es gibt eine handvoll Umgebungen in L^AT_EX, die die Möglichkeiten bieten, vom Blocksatz abzuweichen.

4.3.1 Zentrierter Text

Die Umgebung `center` erzeugt zentrierten Text, indem jegliche Trennung und der Randausgleich ausgeschaltet werden.

```
\begin{center}
<Text>
\end{center}
```

```
8 \begin{center}
9 Dieser Text ist mit der
10 \texttt{center}-Umgebung
11 gesetzt. Ist er l"anger als
12 eine Zeile, entf"allt der
13 Randausgleich.
14 \end{center}
```

Dieser Text ist mit der
center-Umgebung gesetzt. Ist
er länger als eine Zeile, entfällt
der Randausgleich.

[Buch1/SpezTextStruk/center.tex]

Steht die Eingabe bereits innerhalb einer anderen Umgebung, so sollte man statt dessen den Befehl `\centering` verwenden. Damit wird verhindert, dass Zwischenraum durch *zwei aufeinanderfolgende* Umgebungen erzeugt wird.

4.3.2 Einseitig rechts- oder linksbündiger Text

Auf dieselbe Art, wie man zentrierten Text erhält, kann man auch rechts- oder linksbündigen Text erzeugen. Dabei wird an Stelle der `center`-Umgebung `flushright` für rechtsbündigen, und `flushleft` für linksbündigen Text verwendet. Man erhält dadurch Flattersatz auf der linken bzw. rechten Seite des Absatzes.

```
8 \begin{flushleft}
9 Text \texttt{flushleft}
10 \end{flushleft}
11 \begin{flushright}
12 \texttt{flushright}
13 \end{flushright}
```

Text **flushleft**

flushright

[Buch1/SpezTextStruk/ragged.tex]

4.3.3 Zitate

Zitate sind Textteile, die vom laufenden Text durch Abstand und eingezogene Ränder abgehoben werden sollen. Man unterscheidet kurze und lange Zitate; letztere erstrecken sich über mehrere Absätze.

Für kurze Zitate verwendet man die Umgebung `quote`. Innerhalb dieser Umgebung wird der Text ohne Einrückung formatiert. Wird durch `\par` oder Leerzeile ein neuer Absatz erzeugt, dann wird er ebenfalls ohne Einrückung, jedoch mit größerem vertikalem Abstand gesetzt.

```
\begin{quote}
  <Text>
\end{quote}
```

Es wird ein Zwischenraum vor und nach der Umgebung eingefügt. Der rechte und der linke Rand werden um die Längen `\leftmargin` und `\rightmargin` verschoben, d. h. die `quote`-Umgebung wird schmaler als der übrige Text formatiert.

```
8 \begin{quote}
9 Wem das Wasser bis zum Hals
10 steht, der soll den Kopf
11 nicht h"angen lassen.
12
13 Es k"onnte ihm "ubel
14 ergehen \dots{}
15 \end{quote}
```

Wem das Wasser bis
zum Hals steht, der soll
den Kopf nicht h"an-
gen lassen.

Es könnte ihm übel er-
gehen ...

[Buch1/SpecTextStruk/quote.tex]

Für längere Zitate verwendet man die Umgebung `quotation`. Hier werden die Absätze wie im normalen Text gesetzt, d. h. er wird auf den Seiten nicht eingerückt. Bei den Absätzen findet in der ersten Zeile eine Einrückung statt.

```
\begin{quotation}
  <Text>
\end{quotation}
```

Es ist sinnvoll, diese Umgebung lediglich zu verwenden, wenn auch im Haupttext mit Absatzeinrückung gearbeitet wird, um ein einheitliches Aussehen zu gewährleisten.

```

9  ``In turning \TeX{} into \LaTeX{}, I have tried to convert
10 a highly-tuned racing car into a comfortable family sedan.
11 The family sedan isn't meant to go as fast as a racing car
12 or be as exciting to drive, but it's comfortable and gets
13 you to the grocery store with no fuss.''
14
15 \raggedleft
16 Leslie Lamport\\ \LaTeX{}---A Document Preparation System

```

“In turning $\text{\TeX}$ into $\text{\LaTeX}$, I have tried to convert a highly-tuned racing car into a comfortable family sedan. The family sedan isn’t meant to go as fast as a racing car or be as exciting to drive, but it’s comfortable and gets you to the grocery store with no fuss.”

Leslie Lamport
 $\text{\LaTeX}$ —A Document Preparation System

[Buch1/SpezTextStruk/quotation.tex]

4.3.4 Gedichte

In $\text{\LaTeX}$ kann man Verse, Liedertexte oder Gedichte mit Hilfe der Umgebung `verse` schreiben. In dieser Umgebung ist es jedoch erforderlich, explizit den Zeilenumbruch zu steuern.

```

\begin{verse}
  <Text>
\end{verse}

```

Der Zeilenumbruch erfolgt durch die Angabe von `\\` am Ende der Zeile. Strophen werden wie Absätze im normalen Text durch Leerzeilen oder `\par` erzeugt. Geht eine Verszeile über die Ausgabezeile hinaus, bricht $\text{\LaTeX}$ automatisch um und rückt die Fortsetzungszeile etwas ein. Um nach `\\` einen Seitenumbruch zu verhindern, verwendet man statt dessen den Befehl `\\*`.

```

8 \begin{verse}
9 Ein Gedicht, ein Gedicht,
10 ein Gedicht muss her\\
11 ja das ist, ja das ist,
12 ja das ist so schwer.
13
14 Ein Gedicht, ein Gedicht,
15 ein Gedicht muss her\\
16 ja das ist, ja das ist,
17 ja das ist so schwer.
18 \end{verse}
19 \end{document}

```

```

Ein Gedicht, ein Ge-
dicht, ein Gedicht
muss her
ja das ist, ja das ist,
ja das ist so schwer.

Ein Gedicht, ein Ge-
dicht, ein Gedicht
muss her
ja das ist, ja das ist,
ja das ist so schwer.

```

[Buch1/SpezTextStruk/verse.tex]

4.4 Fußnoten und Randnotizen

L^AT_EX verfügt über die Fähigkeit, Anmerkungen zum Text als Fußnote oder Randnote zu formatieren. Fußnoten werden, wie der Name schon andeutet, in der Regel am Fuß einer Seite ausgegeben (außer bei der Umgebung `minipage`, siehe Abschn. 5.3.2, S. 129). Randnoten werden, wie der Name sagt, am Seitenrand auf der Höhe der Stelle gesetzt, an der sie in der Texteingabe definiert wurden.

4.4.1 Fußnoten

Fußnoten (engl. *footnote*) werden erzeugt mit Befehl

```
\footnote{<Text>}
```

Der Befehl muss ohne Leerzeichen direkt nach dem Wort oder Satzzeichen folgen, das eine Fußnotenmarkierung erhalten soll. Der Text erscheint als Fußnote in kleinerer Schrift am Fuß der Seite. Die Markierung der Fußnote, in der Regel eine Nummer, und des Fußnotentexts sind gleich. Die Fußnoten am Ende der Seite werden vom normalen Text durch eine mittellange horizontale Linie getrennt.

```

9   Ein Text mit
10  Fu"snote\footnote{Eine
11  Fu"snote.}.
12
13  Noch ein Satz mit einer
14  Anmerkung.\footnote{Eine
15  zweite Fu"snote.}

```

Ein Text mit Fußnote¹.
 Noch ein Satz mit einer Anmerkung.²

¹Eine Fußnote.
²Eine zweite Fußnote.

[Buch1/SpezTextStruk/footnote1.tex]

Es ist auch möglich, die Nummer der Fußnote anzugeben:

```
\footnote[<Num>]{<Text>}
```

Hier wird das optionale Argument *<Num>* als Fußnotennummer verwendet.

Ein Spezialfall sind Fußnoten in Formeln, Tabellen u.Ä. Umgebungen. Hier kann der Befehl `\footnote` nicht verwendet werden, sondern es muss mit zwei Befehlen gearbeitet werden.

```

\footnotemark[<Num>]
\footnotetext[<Num>]{<Text>}

```

Mittels des Befehls `\footnotemark` wird die Fußnotennummer, d. h. die Markierung der Fußnote, innerhalb der Umgebung gesetzt. Die eigentliche Fußnote wird nach der Umgebung nachgetragen.

```

9   Eine Formel mit
10  Fu"snote
11  \[ x^2_i \textrm{ gegeben}%
12  \footnotemark \]
13  \footnotetext{Eine
14  Fu"snote.}
15
16  Eine Tabelle\\
17  \begin{tabular}{lll}
18  K"ase\footnotemark
19  & Wurst & Quark
20  \end{tabular}
21  \footnotetext{Eine

```

Eine Formel mit Fußnote

$$x_i^2 \text{ gegeben}^1$$

Eine Tabelle
 Käse² Wurst Quark

¹Eine Fußnote.
²Eine zweite Fußnote.

[Buch1/SpezTextStruk/footnote2.tex]

Die Markierung der Fußnoten geschieht standardmäßig mit ganzen (arabischen) Zahlen. Bei den Dokumentklassen `book` und `report` beginnt die Nummerierung bei jedem neuen Kapitel mit 1, bei `article` werden die Fußnoten über den gesamten Artikel durchgezählt.

Soll die Fußnotennummerierung geändert werden, so kann sie mittels

```
\setcounter{footnote}{<Zahl>}
```

geändert werden. Hiermit wird der Fußnotenzähler `footnote` auf den Wert von `<Zahl>` gesetzt.

Im nächsten Beispiel werden zwei Fußnoten gesetzt, eine mit der Nummer 1 und eine mit 6. In einem normalen Text macht dieses Vorgehen natürlich keinen Sinn, es soll hier lediglich Demonstrationszwecken dienen:

```

9  Ein Text mit
10 Fu"snote\footnote{Eine
11 Fu"snote.}.
12
13 \setcounter{footnote}{5}
14
15 Noch ein Satz mit einer
16 Anmerkung.\footnote{Eine
17 zweite Fu"snote.}
```

Ein Text mit Fußnote¹.
Noch ein Satz mit einer Anmerkung.⁶

¹Eine Fußnote.
⁶Eine zweite Fußnote.

[Buch1/SpezTextStruk/footnote3.tex]

Hat man wenige Fußnoten oder möchte man keine Zahlen, so ist es ebenfalls möglich, die Fußnoten mit folgenden Symbolen zu kennzeichnen:

* † ‡ § ¶ || ** †† ‡‡

Die Symbole sind von 1–9 durchnummeriert.

Nun muss der Fußnotenstil geändert werden, damit Symbole statt Zahlen erscheinen:

```
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
```

Jetzt kann man mit dem Befehl

```
\footnote[<Num>]{<Text>}
```

eines der obigen Symbole über die Nummer auswählen, `<Num>` kann nur die Werte von 1 bis 9 annehmen.

Das nächsten Beispiel zeigt eine ganze Ausgabeseite mit Seitennummer am unteren Seitenrand:

```

9  \renewcommand{\thefootnote}
10  {\fnsymbol{footnote}}
11  Ein Text mit
12  Fu"snote\footnote[1]{Eine
13  Fu"snote.}.
14
15  \setcounter{footnote}{6}
16
17  Noch ein Satz mit einer
18  Anmerkung.\footnote{Eine
19  zweite Fu"snote.}
```

Ein Text mit Fußnote*.
Noch ein Satz mit einer Anmerkung.**

*Eine Fußnote.
**Eine zweite Fußnote.

1

[Buch1/SpezTextStruk/footnote4.tex]

Hat der Fußnotenzähler einen Wert größer als 9 oder setzt man für *<Num>* einen solchen ein, erhält man eine entsprechende Fehlermeldung.

4.4.2 Randnotizen

Randnotizen oder auch Marginalien sind Vermerke, die am Rand, im Randbereich neben dem eigentlichen Text, stehen. Für ihre Erzeugung wird der Befehl

```
\marginpar{<Text>}
```

verwendet. Der Text erscheint am äußeren rechten Rand des Dokuments.

```

8  Ein Text mit
9  Randnotiz.\marginpar{Notiz}
10
11  Noch ein Satz mit einer
12  Marginalie.\marginpar{!}
```

Ein Text mit Rand-
notiz. Notiz
Noch ein Satz mit
einer Marginalie. !

[Buch1/SpezTextStruk/margin1.tex]

Mit der Option *twoside*, d.h. insbesondere bei der Dokumentklasse *book*, werden die Randnoten an den äußeren Rändern jedes Blatts gesetzt, also bei geraden Seiten an den linken Rand, bei ungeraden Seiten

an den rechten. Dies kann Probleme machen, wenn die Randnotiz beispielsweise aus einem Pfeil besteht. Dieser müsste je nach Seitennummer eine andere Richtung haben: $\Rightarrow$ für Links und $\Leftarrow$ für Rechts. Der Befehl

```
\marginpar[<L-Text>]{<R-Text>}
```

bietet hier die Lösung. Es wird $\langle L\text{-Text} \rangle$ verwendet, wenn die Marginalie im linken Rand gebildet wird und $\langle R\text{-Text} \rangle$ für den rechten.

```
1 \documentclass{book}
2 \usepackage[T1]{fontenc}
3 \usepackage[ngerman]{babel}
4 %% Längen nur für Buch!
5 \setlength{\textwidth}{3cm}
6 \setlength{\textheight}{4cm}
7 \setlength{\parindent}{0cm}
8 \setlength{\marginparsep}{%
9   {0.3cm}}
10 \setlength{\marginparwidth}{%
11   {1cm}}
12 \pagestyle{plain}
13 \begin{document}
14 \setcounter{page}{4}
15 Ein Text mit
16 Randnotiz.\marginpar{Notiz}
17
18 Noch ein Satz mit einer
19 Marginalie.\marginpar{!}
20 \end{document}
```

Notiz	Ein Text mit Rand-
!	notiz. Noch ein Satz mit einer Marginalie.

4

[Buch1/SpezTextStruk/margin2.tex]

4.5 Gleitobjekte

Bei der Eingabe eines Dokuments steht nicht fest, ob beim Seitenumbruch noch genügend Platz für eine Tabelle oder ein Bild zur Verfügung steht. Besonders unangenehm ist es, wenn am Schluss bei der Feinarbeit durch eine kleine Korrektur plötzlich nicht mehr genügend Platz vorhanden ist, um ein Objekt dort zu lassen, wo es bisher stand. Diese Probleme vermeidet $\text{\LaTeX}$ durch Verwendung von Gleitobjekten. Das können Abbildungen, Tabellen, Fotos u. Ä. sein. Allen gemeinsam ist, dass sie nicht unbedingt an der Stelle in einem Text eingefügt werden müssen, an der sie während der Eingabe geschrieben wurden. Wo sie genau platziert

werden, ergibt sich erst, wenn beim Satz des Dokuments der Umbruch festgelegt wird.

Standardmäßig kennt $\text{\LaTeX}$ die Gleitobjekte

- `figure` für Abbildungen und
- `table` für Tabellen bzw. Tafeln, um den Unterschied zu den Umgebungen `tabular` und `tabbing` hervorzuheben.

Weitere Gleitobjekte können mit dem Makropaket `float` eingerichtet werden.

```
\begin{figure}[<Position>]
  <Abbildung>
\end{figure}
```

bzw.

```
\begin{table}[<Position>]
  <Abbildung>
\end{table}
```

Beide Umgebungen verhalten sich nahezu gleich.

Es gibt auch noch die gesternten Formen, also `table*` und `figure*` für zweispaltige Formatierung. Hierbei wird der Platz über die gesamte Seitenbreite reserviert, so dass breite Objekte gesetzt werden können.

Für das Argument `<Position>` gibt es vier Parameter:

- t** (top) Das Gleitobjekt soll oben, am Anfang einer Seite positioniert werden.
- b** (bottom) Das Gleitobjekt soll unten, am Ende einer Seite positioniert werden.
- p** (page) Die Gleitobjekte werden gesammelt auf Seiten ausgegeben, die nur Gleitobjekte enthalten.
- h** (here) Das Gleitobjekt soll, wenn möglich, an der Stelle ausgegeben werden, wo es in der Texteingabe steht.
- !** Wenn zusätzlich das Argument `!` angegeben wird, beispielsweise `!h`, ignoriert $\text{\LaTeX}$ einige Einschränkungen, die ihm durch Parameter auferlegt sind, und versucht so, den Positionswunsch zu erfüllen.

Die Positions-Parameter sind auch kombinierbar, beispielsweise `tb`. Die Reihenfolge der Parameter, die für `<Position>` angegeben werden können, spielt keine Rolle. Das Argument `bt` bewirkt das gleiche wie `tb`. Es bedeutet also *nicht*, dass zunächst versucht wird, das Objekt auf der Seite oben, und wenn dies nicht möglich ist, auf der Seite unten zu setzen. Es ist nur eine Empfehlung, wohin dieses Gleitobjekt *überhaupt* positioniert werden darf. Wenn keine Angaben gemacht werden, nimmt $\text{\LaTeX}$ die Reihenfolge `tbp` an.

Gerade $\text{\LaTeX}$ -Einsteiger sind mit dem Satz von Gleitobjekten recht unglücklich, da sie es gewohnt sind, einen Text zu schreiben, einen Verweis auf eine Tabelle oder Abbildung zu machen und dieses Objekt sofort im Anschluss positioniert zu sehen. Der Versuch mit der Option `h` scheitert regelmäßig, da die `<Position>` für $\text{\LaTeX}$ nur Empfehlungscharakter hat. Man muss sich mit dem Gleiten der Objekte erst vertraut machen, um zu erkennen, dass dieser Mechanismus unterfüllte Seiten verhindert, die dem Leser sonst einen falschen Eindruck vom anschließenden Text vermitteln. Wenn man ein Objekt unbedingt an eine bestimmte Stelle positionieren möchte, muss man auf das Paket `float` zurückgreifen, welches mit einer neuen Option `H` dies bietet.

Die Befehle `\clearpage`, `\cleardoublepage` und natürlich der Befehl `\end{document}` bewirken, dass alle noch nicht gesetzten Gleitobjekte ausgegeben werden.

Gelegentlich möchte man vermeiden, dass Gleitobjekte auf einer bestimmten Seite ausgegeben werden. $\text{\LaTeX}$ bietet hierzu den Befehl

```
\suppressfloats[<Position>]
```

Ohne Angabe des Arguments `<Position>` wird das nächste ausstehende Gleitobjekt für die laufende Seite verboten. Als Argument für `<Position>` kann entweder `h` oder `b` gewählt werden, um die Ausgabe an den Beginn oder das Ende der Seite zu verhindern. Allerdings können Gleitobjekte, die vor dem Befehl `\suppressfloats` definiert wurden, trotzdem ausgegeben werden.

4.5.1 Layout-Parameter für Gleitobjekte

Werden Layout-Parameter in die Präambel geschrieben, dann gelten sie für das gesamte Dokument. Irgendwo in der Texteingabe gelten Layout-Parameter ab der nächsten Seite für den Rest des Dokuments.

`\topfigrule` | `\botfigrule`

sind Befehle, die eine Trennlinie zwischen Gleitobjekten und dem vorangehenden oder nachfolgenden Text festlegen. Sie sind standardmäßig nicht belegt und können bei Bedarf neu definiert werden. Ein Beispiel:

```
\renewcommand{\topfigure}{ \smallskip
\rule{\textwidth}{2pt}\smallskip }
```

`\topnumber` | `\bottomnumber` | `\totalnumber`

sind Zähler, die die maximal zulässige Anzahl von Gleitobjekten am Kopf bzw. Fuß einer Seite sowie auf der gesamten Seite festlegen. Der Zähler kann bei Bedarf geändert werden. Man schreibt

```
\setcounter{\topnumber}{3}
```

um drei Gleitobjekte am Beginn einer Seite zu erlauben.

`\topfraction` | `\bottomfraction` | `\floatpagefraction`

legen den maximalen Anteil einer Seite fest, die von Gleitobjekten belegt werden dürfen. Einmal für den Beginn der Seite, für das Ende und auf der gesamten Seite.

Diese Parameter können durch Redefinition geändert werden, beispielsweise

```
\renewcommand{\topfraction}{0.5}
```

setzt den Anteil, den Gleitobjekte am Seitenkopf belegen dürfen, auf 0.5, d. h. 50 % der Seite.

`\textfraction`

gibt den Mindestanteil einer Seite an, die von Text belegt sein muss.

`\floatsep` | `\textfloatsep` | `\intextsep`

geben den Abstand eines Gleitobjekts an (als Längenangabe s. 5.4.3) zum Kopf oder Fuß einer Seite, zum Text oder zum Text bei Gleitobjekten, die mit `h` positioniert werden.

`\dblfigrule` | `\dbltopnumber` | `\dbltopfraction` |
`\dblbottomfraction` | `\dblfloatpagefraction`
`\dblfloatsep` | `\dbltextfloatsep`

sind die analogen Parameter für zweispaltiges Layout, `dbl` steht für *double*. Sie haben die oben aufgeführte Bedeutung für einspaltiges Layout.

Hinweis:

Durch Modifikation dieser Parameter lässt sich die Positionierung der Gleitobjekte beeinflussen, meist leider nur verschlechtern. Die Abhängigkeiten der Parameter sind so subtil, dass jede Änderung gut überlegt und sorgfältig ausgetestet werden muss.

4.5.2 Algorithmus zur Ausgabe von Gleitobjekten

$\text{\LaTeX}$ verwendet folgenden Algorithmus zur Ausgabe von Gleitobjekten (aus Goossens et al. (1995)):

- (1) Wenn `!` angegeben wird, werden die meisten Einschränkungen für Gleitobjekte der oben genannten Layout-Parameter ignoriert.
- (2) Wenn `h` angegeben ist, wird versucht, das Gleitobjekte genau an dieser Stelle zu positionieren. Wenn dies nicht möglich ist, dann wird statt dessen mit `t` probiert.
- (3) Bei `t` wird versucht, das Gleitobjekte am Kopf der Seite zu platzieren.
- (4) Wenn `b` angegeben wird, soll versucht werden, das Gleitobjekt an den Fuß der Seite zu platzieren.
- (5) `p` werden alle Gleitobjekte gesammelt, bis eine Seite voll ist, und diese dann auf einer eigenen Seite ausgegeben.
- (6) Die Schritte (3) und (4) werden wiederholt, ebenso (5) solange Gleitobjekte noch nicht ausgegeben sind.

Es gibt Makropakete, die das Arbeiten mit Gleitobjekten besser unterstützen. Zum einen das Paket `float`, welches in einem späteren Band besprochen wird, zum anderen in diesem Band `flatter` (s. 7.5, S. 7.5).

4.5.3 Beschriftung von Gleitobjekten

Gleitobjekte können mit einer Beschriftung versehen werden. $\text{\LaTeX}$ stellt hierzu den Befehl `\caption` bereit.

```
\caption[<Kurztitel>]{<Titel>}
```

Der `{Titel}`, d. h. der Pflichtparameter des Befehls, wird verwendet, um eine Über- oder Unterschrift für das Gleitobjekt zu setzen. Die Position wird durch die Position des Befehls vor oder nach dem Gleitmaterial bestimmt. Standardmäßig sieht $\LaTeX$ vor, dass Titel unter Gleitobjekte gesetzt werden und erzeugt dafür auch vertikalen Abstand vor dem Setzen der Überschrift.

Der `{Kurztitel}` dient nur als Eintrag für das Tabellen- (bei `table`) bzw. Abbildungsverzeichnis (bei `figure`).

Die im Deutschen übliche Tabellenüber- und Abbildungsunterschrift, erreicht man, wie eben gesagt durch Verschieben des `\caption`-Befehls an den Anfang bzw. das Ende der jeweiligen Umgebung `table` bzw. `figure`. Möchte man nun den Abstand der `\caption` vom Gleitobjekt erhöhen, so kann man das analog zum Abschn. 3.6.2 (S. 63) mit dem `@makecaption` tun.

Der Befehl `\caption` setzt den Text „Tabelle“ bzw. „Abbildung“ zusammen mit einer fortlaufenden Nummer vor die Überschrift, die in `\Titel` angegeben ist, sofern das Paket `ngerman` geladen wurde. Standardmäßig verwendet $\LaTeX$ die englischen Begriffe „Table“ und „Figure“. Die Nummer ist in Abhängigkeit von der Dokumentklasse ein oder zweigestuft.

Falls gewünscht kann sowohl von Tabellen als auch von Abbildungen ein Verzeichnis erstellt werden. Hierzu dienen die Befehle:

```
\listoftables
\listoffigures
```

Sie arbeiten ähnlich wie `\tableofcontents`, nur dass die Informationen in Dateien mit der Erweiterung `.lot` bzw. `.lof` geschrieben werden. Der Dateiname ist identisch mit dem Namen der Eingabedatei. Nur zur Erinnerung: Das Dokument muss zweimal von $\LaTeX$ bearbeitet werden, bevor man die Verzeichnisse in der Ausgabe sieht bzw. sie den korrekten Stand widerspiegeln.

Wie beim Inhaltsverzeichnis ist es auch hier möglich, Einträge in die Verzeichnisse von Hand hinzuzufügen. Die Befehle lauten:

```
\addcontentsline{lot}{table}{<Text>}
\addcontentsline{lof}{figure}{<Text>}
```

Um einen Eintrag für ein Verzeichnis zu formatieren, der auch eine Nummer enthält, sollte `<Text>` in der Form

```
\protect\numberline[<Nummer>][<Eintrag>]
```

Nur zur Demonstration, nicht in Realität zu verwenden, ein Beispiel:

```
\addcontentsline{lof}{table}{\protect
\numberline{7.4}{Getürkte Abbildung}}
```

Wieder analog zum Inhaltsverzeichnis besteht die Möglichkeit, Text und Formatierungsbefehle in die Verzeichnisse hineinzubringen. Hierfür existieren auch hier die Befehle:

```
\addtocontents{lot}{<Text>}
\addtocontents{lof}{<Text>}
```

Die eventuell in `<Text>` enthaltenen $\LaTeX$ -Befehle sind zu schützen, da die Information ja zunächst in die Verzeichnisdateien geschrieben und von dort wieder ausgelesen werden. Dass Befehle erst nach dem Einlesen aus der `.lot`- bzw. `.lof`-Datei ausgeführt werden, kann man durch `\protect` erreichen.

4.6 Querverweise

Ein Grund, Objekte wie Tabellen und Abbildungen zu nummerieren, liegt darin, dass man auf diese Weise Bezüge herstellen kann: „Siehe Tabelle 5 auf Seite 54 für weitere Informationen“ ist ein Beispiel dafür. Natürlich gibt es mehr, auf welches man sich beziehen kann, unter anderem Kapitel, Abschnitte, mathematische Formeln, ein nummerierter Aufzählungspunkt, bestimmte Textstellen, die Liste ist endlos.

Solche Querverweise waren in älteren Textverarbeitungsprogrammen mühsam zu erstellen, besonders wenn man sich nicht nur auf Nummern, beispielsweise Kapitelnummern oder Gleichungsnummern, beziehen möchte, sondern außerdem auf die Seitennummer referenziert. Natürlich möchte man auf zurückliegenden Text verweisen, aber auch auf nachfolgenden Text muss dies möglich sein. Mit $\LaTeX$ wird die Erfüllung dieses Wunsches zum Kinderspiel, die Bezüge werden von $\LaTeX$ ermittelt und an den vorgesehen Stellen eingetragen.

Querverweise werden für alle Objekte mit folgenden Befehlen realisiert:

```
\label{<Marke>}
```

Mit diesem Befehl wird eine `{Marke}` an ein Objekt angebracht, auf die von anderen Stellen aus Bezug genommen werden kann.

`\ref{<Marke>}`

hiermit kann, sofern ein Objekt über eine Nummer verfügt, z. B. eine Kapitelnummer, diese Nummer ausgegeben werden.

`\pageref{<Marke>}`

dient dazu die Seitennummer auszugeben, auf der sich das Objekt befindet, welches mit *<Marke>* markiert ist.

```

8 Diese ist eine Auflistung \
9 mit drei Einträgen:
10 \begin{enumerate}
11 \item Eintrag A\label{enum}
12 \item Eintrag B
13 \item Eintrag C
14 \end{enumerate}
15 Siehe \ref{enum} auf
16 Seite \pageref{enum}.
17 \end{document}

```

Diese ist eine Auflistung
mit drei Einträgen:

1. Eintrag A
2. Eintrag B
3. Eintrag C

Siehe 1 auf Seite 1.

[Buch1/SpezTextStruk/quer.tex]

Mit jedem `\label`-Befehl wird eine Marke, die aus Buchstaben, Zahlen und Zeichen gebildet wird, zusammen mit einer Seitennummer und dem Wert eines Zählers in eine Datei mit der Erweiterung `.aux` geschrieben.

Die `\ref`- und `\pageref`-Befehle lesen die erforderliche Information aus der `.aux`-Datei. Dadurch wird es notwendig, mindestens zwei mal `LaTeX` aufzurufen. Eventuell kann ein dritter Lauf vonnöten sein. `LaTeX` meldet dies mit folgendem Hinweis:

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

Eigene Textstrukturen

5.1 Schriften

Ein gutes, typografisch korrekt gesetztes Dokument zeichnet sich durch das harmonische Zusammenspiel verschiedener Schriften aus. Ein Hauptfehler, der oft gemacht wird, ist die Verwendung vieler – meist nicht zusammenpassender – Schriften. Professionelle Typografie verwendet zwei, drei maximal vier Schriften in einem Text (natürlich kein Werbetext).

L^AT_EX beachtet diese Regeln und verwendet nur Schriften einer Schriftfamilie. Um hiervon abzuweichen, muss ein guter Grund vorhanden sein.

5.1.1 T_EX-Zeichensatz

Innerhalb von T_EX ist ein Zeichensatz (engl. *font*) eine Zusammenstellung von 128 Zeichen. Die Zuordnung, d. h. die Anordnung der Zeichen im Zeichensatz wird als Kodierung (engl. *encoding*) bezeichnet. Zum T_EX-System zählen als Standardzeichensatz die ‚Computer Modern‘-Schriften, meist kurz als CM-Fonts bezeichnet.

Mit der Version T_EX 3.0 wurde der Zeichensatz erweitert, um 256 Zeichen unterstützen zu können. Hierfür wurde auf der Basis der CM-Fonts die EC-Fonts entwickelt – die Fonts mit den europäischen Sonderzeichen unter dem Namen ‚European Modern‘.

Innerhalb eines Zeichensatzes ist jedes einzelne Zeichen durch eine Zahl (d. h. 0–127 für CM-Fonts, 0–255 für EC-Fonts) aufrufbar. Der Befehl hierfür lautet

	'0	'1	'2	'3	'4	'5	'6	'7	
'00x	Γ	Δ	Θ	Λ	Ξ	Π	Σ	Υ	"0x
'01x	Φ	Ψ	Ω	ff	fi	fl	ffi	ffl	
'02x	ı	ı	ı	ı	ı	ı	ı	ı	
'03x	ı	ß	æ	œ	o	Æ	Œ	Œ	"1x
'04x	ı	!	"	#	\$	%	&	'	
'05x	(	)	*	+	.	-	.	/	"2x
'06x	0	1	2	3	4	5	6	7	
'07x	8	9	:	:	i	=	i	?	"3x
'10x	@	A	B	C	D	E	F	G	
'11x	H	I	J	K	L	M	N	O	"4x
'12x	P	Q	R	S	T	U	V	W	
'13x	X	Y	Z	[	"	]	ı	'	"5x
'14x	ı	a	b	c	d	e	f	g	
'15x	h	i	j	k	l	m	n	o	"6x
'16x	p	q	r	s	t	u	v	w	
'17x	x	y	z	ı	ı	ı	ı	ı	"7x
	"8	"9	"A	"B	"C	"D	"E	"F	

Abbildung 5.1: CM-Font und Kodierung

```
\symbol{<Nummer>}
```

wobei *<Nummer>* als Dezimalzahl, Oktalzahl oder als Hexadezimalzahl angegeben werden kann. (Beispielsweise würde das große ‚A‘ mit dezimal 65 oder oktal '101 oder hexadezimal "49 ausgewählt.) Bei der Bearbeitung eines Textes durch $\text{\LaTeX}$ sind nur die Maße der Zeichen relevant. Für $\text{\LaTeX}$ und speziell für $\text{\TeX}$, das den Satz vornimmt, ist ein Buchstabe eine Box mit festgelegter Breite und Höhe (diese setzen sich aus Ober- und Unterlängen zusammen), die auf der Grundlinie positioniert wird.

5.1.2 $\text{\LaTeX}$ -Zeichensatz

Während man bei der Arbeit mit $\text{\TeX}$ die Zeichensätze explizit auswählen muss, ist dies bei $\text{\LaTeX}$ eleganter geregelt.

Bei $\text{\LaTeX}$, genauer bei $\text{\LaTeX}2_{\epsilon}$ mit NFSS (*new font selection scheme*), bestimmen fünf voneinander unabhängige Schriftattribute das Aussehen des Dokuments. Diese sind:

- Schriftfamilie
- Schriftform
- Schriftstärke (Schriftbreite)
- Schriftgröße (Schriftgrad)
- Kodierung

5.1.2.1 Schriftfamilie

$\LaTeX$ kennt standardmäßig drei Schriftfamilien:

- Serifen-Schriften
- Serifenlose-Schriften
- Schreibmaschinenschrift (Äquidistanz-Schrift)

Serifen sind Füßchen und Häkchen an den einzelnen Buchstaben, die die Schrift leichter lesbar machen, da sie das Auge an der Zeile entlangführen. Wer viel liest, wird diese Schriftfamilie zu schätzen wissen. Serifenlose Schriften eignen sich dagegen mehr für Plakate u. Ä. Bei Schreibmaschinenschriften ist jedes Zeichen gleich breit, wie es früher und auch noch heute bei Schreibmaschinen war und ist. Damit eignet sich diese Schrift weniger zum Setzen von gut lesbaren Dokumenten, sie ist jedoch beim Auflisten von Programmcode hervorragend geeignet, da sie eine klare Anordnung hat. Ein Beispiel dafür ist dieses Buch.

Innerhalb von $\LaTeX$ kann man auf diese Schriftfamilien wie folgt zugreifen:

Tabelle 5.1: Schriftfamilien

Schrift	Deklaration	Beispiel
Serifen-Schrift	<code>\rmfamily</code>	Das ist ein Probe-Text
Serifenlose-Schrift	<code>\sffamily</code>	Das ist ein Probe-Text
Schreibmaschinenschrift	<code>\ttfamily</code>	Das ist ein Probe-Text

5.1.2.2 Schriftform

Schriftform ist das zweite Schriftattribut, das unabhängig von den anderen änderbar ist. In der Regel wird für Dokumente eine senkrechte Form verwendet. Wichtig ist dann der Wechsel in eine Kursivschrift oder auf Kapitälchen (kleine Großbuchstaben), um Textteile hervorzuheben. Zusätzlich zur Kursivschrift kennt $\LaTeX$ auch noch eine Schrägschrift.

Tabelle 5.2: Schriftformen

Schrift	Deklaration	Beispiel
aufrechte Schrift	<code>\upshape</code>	Das ist ein Probe-Text
kursive Schrift	<code>\itshape</code>	<i>Das ist ein Probe-Text</i>
schräge Schrift	<code>\slshape</code>	<i>Das ist ein Probe-Text</i>
Kapitälchen	<code>\scshape</code>	DAS IST EIN PROBE-TEXT

5.1.2.3 Schriftstärke

Mit dieser Ausprägung wird die Dicke, also die Breite eines einzelnen Buchstabens, und die Strichstärke der Zeichen einer Schrift beschrieben. Standardmäßig sind zwei Strichstärken definiert.

Tabelle 5.3: Schriftstärke

Schrift	Deklaration	Beispiel
normale Stärke	<code>\mdseries</code>	Das ist ein Probe-Text
fette Stärke	<code>\bfseries</code>	Das ist ein Probe-Text

5.1.2.4 Schriftgröße

Jede Schrift ist in unterschiedlichen Größen vorhanden und falls nicht, kann sie erzeugt werden. Die Größe wird in Punkten (pt) angegeben, wobei 1 pt ungefähr 1/3 mm ist. Die Schriftgröße ist kein absolutes Maß, sondern sie gibt vielmehr einen Richtwert für den Benutzer an. Des Weiteren können zwei gleichgroße Schriften nicht immer kombiniert werden, da ihr Erscheinungsbild zueinander passen muss.

Traditionsgemäß sind, von $\text{\TeX}$ übernommen, Schriftgrößen vorhanden, die durch Multiplikation der Basisgrößen mit dem Faktor 1.2 entstehen. Ist die Basisgröße 10 pt, dann ergeben sich die Größen 12 pt, 14.4 pt, 17.28 pt usw.

5.1.2.5 Kodierung

Die Kodierung legt die Reihenfolge der Zeichen innerhalb eines Zeichensatzes fest. Normalerweise wird die Kodierung in der Präambel festgelegt, indem das Makropaket `fontenc` geladen wird.

```
\usepackage[<Kodierung>]{fontenc}
```

Das optionale Argument *<Kodierung>* kann eine durch Komma getrennte Liste von Kodierungen enthalten. Im Text könnte dann an beliebiger Stelle zum wechseln der Kodierung die Befehlsfolge:

```
\fontencoding{<Kodierung>}
\selectfont
```

verwendet werden.

Gängige Kodierungen sind:

Tabelle 5.4: Kodierungen

T1	8-Bit-Standard für EC-Fonts
TS	8-Bit-Zeichenkodierung für TS-Fonts
T1	8-Bit-Standard für EC-Fonts
OT1	7-Bit-Kodierung für Text ¹ , CM-Fonts
OML	7-Bit-Kodierung für Formeltext ¹
OMS	7-Bit-Kodierung für mathematische Symbole ¹
OMX	7-Bit-Kodierung für erweiterte mathematische Symbole ¹

Bei diesem Buch wird das Makropaket fontenc mit folgenden Kodierungen geladen.

```
\usepackage[TS1,TS]{fontenc}
```

5.1.2.6 Zeichensatzbefehle

Die nachfolgenden `\text . . .`-Befehle werden verwendet, um die Schrift bei kurzen Textstücken zu ändern.

```
\textxx{<Text>}
```

Im Argument *<Text>* darf kein Absatz vorkommen, d. h. keine Leerzeile und kein `\par`-Befehl.

Die folgende Tabelle gibt eine Übersicht über die möglichen Kombinationen:

¹ Kodierungsempfehlung von D. E. Knuth, dem Autor von $\text{\TeX}$.

Tabelle 5.5: \text...-Befehle

Befehl	entspricht	Beispiel
\textrm{...}	{\rmfamily ...}	Das ist ein Probe-Text
\textsf{...}	{\sffamily ...}	Das ist ein Probe-Text
\texttt{...}	{\ttfamily ...}	Das ist ein Probe-Text
\textup{...}	{\upshape ...}	Das ist ein Probe-Text
\textit{...}	{\itshape ...}	<i>Das ist ein Probe-Text</i>
\textsl{...}	{\slshape ...}	<i>Das ist ein Probe-Text</i>
\textsc{...}	{\scshape ...}	DAS IST EIN PROBE-TEXT
\textmd{...}	{\mdseries ...}	Das ist ein Probe-Text
\textbf{...}	{\bfseries ...}	Das ist ein Probe-Text
\textnormal{...}	{\normalfont ...}	Das ist ein Probe-Text

5.1.2.7 Deklarationsbefehle

Neben kurzen Textstücken können auch längere Textblöcke, die sich über mehrere Absätze erstrecken, bis zum gesamten Dokument in einer anderen Schrift gesetzt werden. In der Präambel verwendet, gilt die gewählte Schrift für das gesamte Dokument. Ansonsten wirkt der Befehl ab der betreffenden Stelle. Die Wirkung eines Deklarationsbefehls kann durch einen Block begrenzt werden.

Tabelle 5.6: Deklaration von Schriften

Befehl	entspricht	Beispiel
{\rmfamily ...}	\textrm{...}	Das ist ein Probe-Text
{\sffamily ...}	\textsf{...}	Das ist ein Probe-Text
{\ttfamily ...}	\texttt{...}	Das ist ein Probe-Text
{\upshape ...}	\textup{...}	Das ist ein Probe-Text
{\itshape ...}	\textit{...}	<i>Das ist ein Probe-Text</i>
{\slshape ...}	\textsl{...}	<i>Das ist ein Probe-Text</i>
{\scshape ...}	\textsc{...}	DAS IST EIN PROBE-TEXT
{\mdseries ...}	\textmd{...}	Das ist ein Probe-Text
{\bfseries ...}	\textbf{...}	Das ist ein Probe-Text
{\normalfont ...}	\textnormal{...}	Das ist ein Probe-Text

```

8 Das ist ein Probe-Text.
9 \textbf{Das ist ein
10 Probe-Text.}
11 Das ist ein Probe-Text.
```

Das ist ein Probe-Text. **Das ist ein Probe-Text.** Das ist ein Probe-Text.

5.1.2.8 Schriftumgebungen

Für jeden Deklarationsbefehl gibt es zusätzlich die Möglichkeit, den Befehlsnamen (ohne Backslash \) als Umgebungsnamen zu verwenden.

Tabelle 5.7: Schriftumgebungen

<code>\begin{rmfamily}</code>	...	<code>\end{rmfamily}</code>
<code>\begin{sffamily}</code>	...	<code>\end{sffamily}</code>
<code>\begin{ttfamily}</code>	...	<code>\end{ttfamily}</code>
<code>\begin{upshape}</code>	...	<code>\end{upshape}</code>
<code>\begin{itshape}</code>	...	<code>\end{itshape}</code>
<code>\begin{slshape}</code>	...	<code>\end{slshape}</code>
<code>\begin{scshape}</code>	...	<code>\end{scshape}</code>
<code>\begin{mdseries}</code>	...	<code>\end{mdseries}</code>
<code>\begin{bfseries}</code>	...	<code>\end{bfseries}</code>
<code>\begin{normalfont}</code>	...	<code>\end{normalfont}</code>

Im Gegensatz zu vielen anderen Umgebungen wird vor und nach einer Schriftumgebung kein Leerraum eingefügt, d. h. auch keine Absätze. Daher können diese Umgebungen an jeder Stelle, also auch mitten im Text stehen.

```

8 Das ist ein Probe-Text.
9 \begin{bfseries}
10 Das ist ein Probe-Text.
11 Das ist ein Probe-Text.
12 Das ist ein Probe-Text.
13 \end{bfseries}
14 Das ist ein Probe-Text.
```

Das ist ein Probe-Text. **Das ist ein Probe-Text. Das ist ein Probe-Text. Das ist ein Probe-Text.** Das ist ein Probe-Text.

[Buch1/EigTextStrukt/bfseriesum.tex]

5.1.2.9 Wechsel der Schriftgrößen

Die Änderung der Schriftgröße wird mit den folgenden Befehlen vorgenommen:

Tabelle 5.8: Schriftgrößen

Schriftgröße	Deklaration	Beispiel
winzig	<code>\tiny</code>	winzig
sehr klein	<code>\scriptsize</code>	sehr klein
kleiner	<code>\footnotesize</code>	kleiner
klein	<code>\small</code>	klein
normal	<code>\normalsize</code>	normal
groß	<code>\large</code>	groß
größer	<code>\Large</code>	größer
sehr groß	<code>\LARGE</code>	sehr groß
riesig	<code>\huge</code>	riesig
gigantisch	<code>\Huge</code>	gigantisch

So hintereinandergeschrieben sieht das natürlich besonders scheußlich aus. Neben der Schriftgröße, muss man auch dafür Sorge tragen, dass der Zeilenabstand passend ist.

Die Schriftgrößen-Befehle verhalten sich wie die Deklarationsbefehle und gelten bis zum Ende des Blocks, indem sie eingesetzt wurden, oder bis ein anderer Schriftgrößen-Befehl folgt.

```

8  Das ist ein Probe-Text.
9  {\Large
10  Das ist ein Probe-Text.
11  Das ist ein Probe-Text.
12  Das ist ein Probe-Text.}
13 Das ist ein Probe-Text.
```

Das ist ein Probe-Text. Das ist ein Probe-Text. Das ist ein Probe-Text. Das ist ein Probe-Text. Das ist ein Probe-Text.

[Buch1/EigTextStrukt/large.tex]

Anstelle der Größenbefehle kann man, wie bei den Deklarationen, eine Umgebung mit gleichem Namen verwendet werden.

5.1.2.10 Kombination verschiedener Schriftbefehle

Wie schon früher verwendet, können die verschiedenen Deklarationen und Schriftbefehle kombiniert werden. Dadurch kann eine Schrift gewählt werden, auf die alle Attribute zutreffen:

```

8 Das ist ein Probe-Text,
9 der teilweise in
10 {\sffamily\itshape%
11 \large gro"ser kursiver
12 serifenlosen Schrift} und
13 teilweise in
14 {\sffamily\bfseries%
15 \small kleiner fatter
16 Schreibmaschinenschrift}
17 gesetzt wird.

```

Das ist ein Probe-Text, der teilweise in *großer kursiver serifenlosen Schrift* und teilweise in **kleiner fatter Schreibmaschinenschrift** gesetzt wird.

[Buch1/EigTextStrukt/schkombi.tex]

5.1.3 Hervorhebung von Text

Um Text hervorzuheben gibt es in $\text{\LaTeX}$ den Befehl `\emph`. Mit ihm wird der Text normalerweise kursiv gesetzt.

```
\emph{<Text>}
```

Der Befehl ist, wie auch die `\text...`-Befehle nur für kurze Textteile geeignet. Für längere Texte gibt es die Umgebung `em`.

```

8 Das ist \emph{hervorgehobener}
9 Text, oder
10
11 \begin{em}
12 ein l"angeres Textst"uck, das
13 sogar eine
14
15 Leerzeile enthalten kann,
16 d.\,h. es besteht
17 \end{em}
18
19 aus zwei Paragraphen.

```

Das ist *hervorgehobener* Text, oder ein längeres Textstück, das sogar eine Leerzeile enthalten kann, d. h. es besteht aus zwei Paragraphen.

[Buch1/EigTextStrukt/emtext.tex]

Andere Möglichkeiten sind Auszeichnungen mit fatter und schräger Schrift. Als Relikt aus dem Schreibmaschinenzeitalter ist es auch möglich, Textteile zu unterstreichen. Hierfür dient der Befehl `\underline`.

```
\underline{<Text>}
```

Leider führt der Befehl dazu, dass die unterstrichenen Textteile nicht mehr getrennt werden können. Daraus folgt, dass maximal eine Textzeile im gesetzten Dokument unterstrichen werden kann.

```

8  Das ist
9  \underline{hervorgehobener}
10 Text.

```

Das ist hervorgehobener Text.

[Buch1/EigTextStrukt/unttext.tex]

Eine Hervorhebung durch Unterstreichen sollte man unterlassen, es gibt bessere Möglichkeiten (s. o. Fettschrift).

5.2 Akzente und Sonderzeichen

Andere Sprachen, neben den schon im vorigen Kapitel erwähnten Englisch und Deutsch, haben ebenfalls eine Auswahl von Akzenten und weiteren speziellen Zeichen.

5.2.1 Akzente

L^AT_EX kennt eine große Vielfalt von Akzenten, die über oder unter einen (oder mehrere) Buchstaben positioniert werden können. Einzig die Buchstaben i und j benötigen eine Sonderregelung, da hier der Akzent den Punkt auf dem Buchstaben ersetzt. Sie werden dann durch \i bzw. \j kodiert.

In L^AT_EX gibt es die folgenden Akzente:

```

8  \={o} \. {o} \u{o}   \\
9  \v{o} \r{o} \H{o}   \\
10 \d{o} \b{o} \k{o}   \\
11 \c{o} \t{oo}

```

ō ó ö
 ő ô õ
 ȝ ȡ Ȣ
 Ȣ ȢȢ

[Buch1/EigTextStrukt/acc1.tex]

Damit kann man beispielsweise folgenden Text schreiben

```

8  El se\~nor est\'a bien,
9    gar\c{c}on.\.
10 \'El est\'a aqu\'i.

```

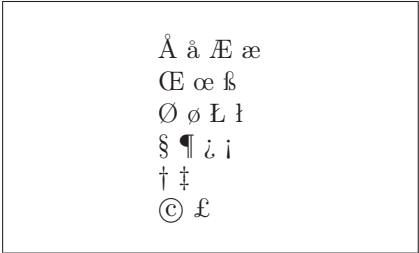
El señor está bien, garçon.
Él está aquí.

[Buch1/EigTextStrukt/acc2.tex]

5.2.2 Weitere Sonderzeichen

Eine ganze Anzahl von Spezialzeichen steht ebenfalls zur Verfügung, beispielsweise die in Skandinavien gebräuchlichen Buchstaben \O, \o oder \AE, \ae:

```
8 \AA{} \aa{} \AE{} \ae{} \\
9 \OE{} \oe{} \ss{} \\
10 \O{} \o{} \L{} \l{} \\
11 \S{} \P{} ? ' ! ' \\
12 \dag{} \ddag{} \\
13 \copyright{} \pounds{}
14 \end{document}
```



[Buch1/EigTextStrukt/acc3.tex]

Es gibt noch weitere fremdsprachige Akzente, die jedoch nur im mathematischen Modus verwendet werden können (vgl. 6.3.11, S. 167).

5.2.3 Angloamerikanische Anführungszeichen

Für die amerikanischen, genauer im englischsprachigen Raum verwendeten Versionen der Anführungszeichen in “Wort” und ‘Wort’ gibt es die Befehle:

```
\textquotedblleft \textquotedblleft
\textquoteleft \textquoteleft
```

Den ersten Befehl kann man zur besseren Klarheit lesen als text-quote-double-left, analog gilt dies für die anderen. Die Kurzform lautet:

```
\` \' \' \'
```

Diese sind noch einmal in Tabelle 5.9 zusammengefasst.

Tabelle 5.9: Angloamerikanische Anführungszeichen

Befehl	Kurzform	Beschreibung
\textquotedblleft	\`	angloam. linkes doppeltes Anführungszeichen
\textquotedblright	\'	angloam. rechtes doppeltes Anführungszeichen
\textquoteleft	\`	angloam. linkes einfaches Anführungszeichen
\textquoteright	\'	angloam. rechtes einfaches Anführungszeichen

5.2.4 Bindestriche, Gedankenstriche und Andere

In deutschsprachigen Texten kommen, abgesehen vom Minuszeichen, zwei verschiedene Striche vor. Der Trenn- oder Bindestrich ist immer kurz »-«, der Gedankenstrich ist länger »-«. Die Eingabe:

```
-      --
```

Als Faustregel gilt: Werden zusammengesetzte Wörter mit Eigennamen gebildet, werden die einzelnen Teile mit Bindestrich verbunden (Genaueres regelt der Duden, Dudenredaktion (1996)). Da der Bindestrich für L^AT_EX ein Sonderzeichen darstellt, kann danach keine Silbentrennung mehr erfolgen. Insbesondere bei kurzen Wörtern nach dem Strich ist das auch gut, denn sonst könnte der Text unlesbar werden. Bei deutschen Wortungetümen wie „Eisenbahner-Gewerkschaftshaus“ ist dies unerwünscht, daher sollte man hier die Eingabe des - durch ein ~= ersetzen. Andererseits gibt es auch Bindestriche, bei denen partout nicht getrennt werden soll wie „T-Punkt“ oder bei Aufzählungen, was mit "~ vermieden wird.

Ordentlicherweise unterscheidet man auch Gedankenstrich und Von-Bis-Strich, die Länge des Strichs ist gleich, beide werden mit -- eingegeben, nur beim Gedankenstrich gehört noch Leerraum links und rechts vom Strich drumherum; im Gegensatz zum Angloamerikanischen.

```
8 Trenn- oder Bindestrich
9 Heinrich-von-Kleist-Str.~
10 Eisenbahner~=Gewerkschaftshaus
11
12 x"~beliebig\\
13 bergauf und "~ab\\
14 I"~, i- und T"~Punkt
15
16 A--Z \\
17 1--9 \\
18 Heidelberg-- Mannheim
```

```
Trenn- oder Bindestrich Heinrich-
von-Kleist-Str. Eisenbahner-Ge-
werkschaftshaus
x-beliebig
bergauf und -ab
I-, i- und T-Punkt
A-Z
1-9
Heidelberg – Mannheim
```

[Buch1/EigTextStrukt/strich.tex]

Das aktive Zeichen Tilde ~ verhindert einen Zeilenumbruch nach „Heidelberg“, so dass der Gedankenstrich nicht zu Beginn einer neuen Zeile steht, das würde den Lesefluss stören.

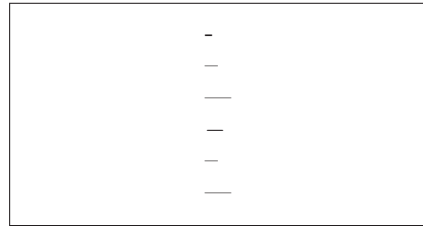
Im englischen Sprachraum kommt in der Regel der ganz lange Gedankenstrich ohne Leerraum zum Einsatz »—«, zum Beispiel *Yes—or no?*.

Zusätzlich stellt $\text{\LaTeX}$ neben dem Minusstrich, der im mathematischen Modus eingegeben werden muss, die Befehle

```
\textendash
\textemdash
```

bereit.

```
9  -          \\\
10 --         \\\
11 ---        \\\
12 $-$        \\\
13 \textendash \\\
14 \textemdash
```



[Buch1/EigTextStrukt/strich2.tex]

5.2.5 Erweiterung zur Trennung

Manchmal möchte man verhindern, dass eine Trennung oder ein Zeilenumbruch erfolgt. Die Trennung kann man exakt festlegen (s. o.), wenn man die Trennstellen mit `\-` markiert. Wenn man jedoch überhaupt keine Trennung wünscht, hilft dieser Befehl leider nicht. Statt dessen kann man auf den Befehl `\mbox` zurückgreifen.

```
\mbox{<Text>}
```

Hier wird verhindert, dass das Argument `<Text>` getrennt werden kann.

Es kann auch der Fall sein, dass an einem Leerzeichen kein Zeilenumbruch erfolgen soll. Hier muss das Leerzeichen durch ein geschütztes Leerzeichen werden, die Eingabe erfolgt durch das Tilde-Zeichen

```
~
```

Eine `~` (Tilde) zu schreiben ist das Gleiche wie ein Leerzeichen einzugeben, mit dem Unterschied, dass kein Zeilenumbruch möglich ist. Man sollte jedoch darauf achten, dass zusätzlich zur `~` kein weiteres Leerzeichen geschrieben wird, es würde zusätzlich hinzugefügt. Ebenso werden, anders als beim Leerzeichen, mehrere Tilden nicht zu einem Leerraum zusammengefasst.

```

8 Prof.~Dr.~Donald E.~Knuth
9
10 Prof.~ Dr.~ Donald E.~ Knuth
11
12 Prof.~~~Dr.~~~Donald~~~%
13 E.~~~Knuth

```

Prof. Dr. Donald E. Knuth Prof. Dr. Donald E. Knuth Prof. Dr. Donald E. Knuth

[Buch1/EigTextStrukt/mbox.tex]

5.3 Boxen

Boxen sind die Grundlage des Satzsetzes, den $\text{T}_{\text{E}}\text{X}$ als Satzmaschine im Hintergrund durchführt (siehe 1.1 auf Seite 4). $\text{T}_{\text{E}}\text{X}$ baut eine Seite aus Boxen zusammen. Man kann auch sagen, $\text{T}_{\text{E}}\text{X}$ und damit auch $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ ‚denkt‘ in Boxen. $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ verwendet Boxen sehr häufig, um Dinge zusammenzufassen, die als Einheit weiterbearbeitet werden sollen. Daher ist eine Box ein Objekt, das sich gegenüber $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ bzw. $\text{T}_{\text{E}}\text{X}$ als ein einzelnes Zeichen verhält. Eine Box kann nicht geteilt, oder am Zeilen- oder Seitenrand umbrochen werden.

$\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ kennt drei verschiedene Box-Typen:

Horizontale Boxen (auch LR-Boxen genannt) Hier werden die Einzelbestandteile von links nach rechts angeordnet. Zeichen werden zu einer Einheit zusammengefasst.

Absatzboxen sind Boxen, die aus Zeilen aufgebaut und (fast) wie ein Absatz gesetzt werden. Das bedeutet, dass in diesen Boxen ein Zeilenumbruch möglich ist.

Rechteckboxen sind mit Farbe gefüllte Rechtecke bzw. horizontale oder vertikale Linien.

5.3.1 Horizontale Boxen

Mit folgenden Befehlen kann man einfache horizontale Boxen definieren:

```

\mbox{<Text>}
\makebox[<Breite>][<Pos>]{<Text>}

```

`\mbox` erzeugt eine Box, deren Breite und Höhe durch den in ihr stehenden Text bestimmt wird. Eine solche Box kann man auch verwenden, wenn ein Wort nicht getrennt werden soll.

`\makebox` bietet die Möglichkeit, die Breite der Box sowie die Positionierung des darin enthaltenen Textes zu bestimmen. Es wird eine Box in der angegebenen *<Breite>* erstellt und der in ihr stehende Text entsprechend der Position *<Pos>* angeordnet. Für *<Pos>* bedeutet der Buchstaben *l*, den Text linksbündig und ein *r* ihn rechtsbündig in die Box zu positionieren. Wird das Argument nicht verwendet, so wird der Text innerhalb der Box zentriert. Mit der Angabe *s* kann der Text vom linken bis zum rechten Rand gedehnt werden, allerdings nur unter der Voraussetzung, dass dehnbarer Leerraum in der Box verfügbar ist.

```

8   Telefonnummer
9   \mbox{06\,21/12\,34\,56}\\
10  \makebox[3cm]{Wichtig!}\\
11  Der Inhalt dieser Box ist
12  \makebox[1.4cm][l]%
13    {\emph{links-}},
14  dieser
15  \makebox[1.4cm][r]%
16    {\emph{rechts}}b"undig.
```

Telefonnummer 06 21/12 34 56

Wichtig!

Der Inhalt dieser Box ist *links-*,
dieser *rechtsbündig*.

[Buch1/EigTextStrukt/ebox.tex]

Um horizontale Boxen einzurahmen, d. h. den Rand der Box durch eine Linie auch optisch sichtbar zu machen, sind folgende Befehle von Nutzen:

```

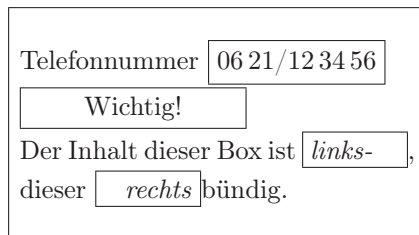
\fbbox{<Text>}
\framebox[<Breite>][<Pos>]{<Text>}
```

Die Arbeitsweise ist analog zu den einfachen Boxen `\mbox` und `\makebox`. Es werden die gleichen Argumente erwartet. Die Möglichkeit, einen Rahmen um eine Box zu legen, ist sehr nützlich, wenn man optisch sehen möchte, wieviel Platz ein Objekt auf der Seite belegt. Im normalen Text wird man in der Regel mehr die Befehle `\mbox` und `\makebox` verwenden.

```

8  Telefonnummer
9  \fbox{06\,21/12\,34\,56}\\
10 \framebox[3cm]{Wichtig!}\\
11 Der Inhalt dieser Box ist
12 \framebox[1.4cm][l]%
13   {\emph{links-}},
14 dieser
15 \framebox[1.4cm][r]%
16   {\emph{rechts}}b"undig.

```



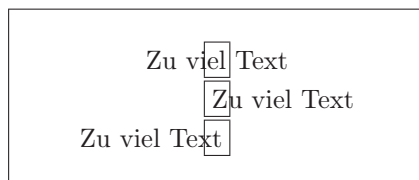
[Buch1/EigTextStrukt/efbox.tex]

Legt man eine Box mit fester Breite an und der Text passt nicht in die Box, so wird je nach Wahl des Arguments *<Pos>* über die Box hinausgeschrieben. Das ist kein Fehler, sondern kann bei der Positionierung von Text geschickt eingesetzt werden. Ist als Argument ein *l* verwendet, dann wird linksbündig positioniert und damit nach rechts geschrieben. Bei Angabe eines *r* entsprechend nach links. Keine Angabe bedeutet, dass der Text rechts und links die Box überragt, was allerdings weniger von Bedeutung ist.

```

8  \framebox[1em]%
9    {Zu viel Text}\\
10 \framebox[1em][l]%
11   {Zu viel Text}\\
12 \framebox[1em][r]%
13   {Zu viel Text}

```



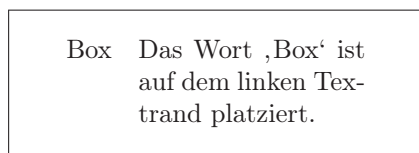
[Buch1/EigTextStrukt/eoverfbox.tex]

Mit einer Box kann man alles Mögliche tun, beispielsweise auch Text an den Seitenrand positionieren und so eine Randmarkierungen vortäuschen.

```

8  \makebox[0pt][r]{Box\ \ \ }%
9  Das Wort \glq{}Box\grq{}
10 ist auf dem linken
11 Textrand platziert.

```



[Buch1/EigTextStrukt/emakebox.tex]

Der Trick besteht hier darin, eine Box der Breite 0 direkt am Seitenrand zu positionieren, und dann den Text entweder links am linken oder recht am rechten Seitenrand hinauszuschreiben.

5.3.1.1 Box-Parameter

Bei `\fbox` und `\framebox` können zwei Parameter verändert werden:

`\fboxrule`

bestimmt die Liniendicke der Rahmen.

`\fboxsep`

bestimmt den Leerraum zwischen Rahmen und Text.

```

8 \addtolength{\fboxrule}{1mm}
9 \fbox{Piet"at}
10 \addtolength{\fboxrule}{-1mm}
11 \setlength{\fboxsep}{1mm}

```



[Buch1/EigTextStrukt/efboxparm.tex]

5.3.1.2 Boxen speichern

Will man die gleiche `\mbox` oder `\makebox` an mehreren Stellen verwenden, so kann man sie speichern. Das hat den Vorteil, dass sie nur einmal definiert wird, am Besten in der Präambel, und dann über den definierten Befehlsnamen abgerufen werden kann. Die notwendigen Befehle lauten:

```

\newsavebox{\<Name>}
\sbox{\<Name>}{\<Text>}
\savebox{\<Name>}[<Breite>][<Pos>]{\<Text>}
\usebox{\<Name>}

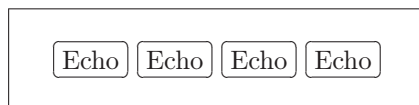
```

1. Zunächst wird mit `\newsavebox` eine Box benannt. `<Name>` kann vom Benutzer frei gewählt werden.
2. Dann wird mit dem Befehl `\sbox` oder `\savebox` eine Box mit dem Inhalt `<Text>` erzeugt und gespeichert.
3. Nun kann die Box `<Name>` an jeder beliebigen Stelle aufgerufen werden.

```

7 \usepackage{fancybox}
8 \begin{document}
9 \newsavebox{\Odw}
10 \sbox{\Odw}%
11   {\ovalbox{Echo}}
12 \usebox{\Odw} \usebox{\Odw}
13 \usebox{\Odw} \usebox{\Odw}
14 \end{document}

```



[Buch1/EigTextStrukt/esavebox.tex]

Der Befehl `\ovalbox` stammt aus dem Paket `fancybox`.

5.3.1.3 Boxen vertikal verschieben

Mit dem Befehl

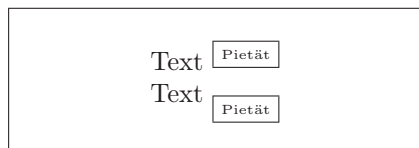
```
\raisebox{<Schieb>}[<OberL>][<UnterL>]{<Text>}
```

wird eine Box vom Typ `\mbox` erzeugt, die den Text `<Text>` enthält. Zusätzlich wird die Box um `<Schieb>` nach oben bzw. bei negativem Wert nach unten verschoben. Das Argument `OberL` bestimmt die Höhe der Box oberhalb der Grundlinie, `UnterL` die Höhe unterhalb der Grundlinie. Ohne diese Angaben nimmt $\text{\LaTeX}$ die sich aus dem Inhalt der Box ergebenden Werte.

```

8 Text
9 \raisebox{1ex}%
10   {\tiny\fbbox{Piet"at}}
11
12 Text
13 \raisebox{-1ex}%
14   {\tiny\fbbox{Piet"at}}

```



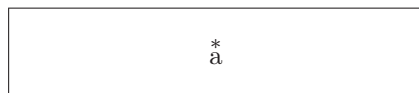
[Buch1/EigTextStrukt/erbox.tex]

Diesen Mechanismus kann man auch verwenden, wenn man ein Zeichen über ein anderes setzen will. Allerdings sollte man vorher in die Tabellen des Pakets `textcomp` schauen, ob es dieses Zeichen nicht schon gibt, bevor man sich diese Mühe macht.

```

8 a\hspace{-.16cm}%
9   \raisebox{.12cm}%
10   {\footnotesize *}

```



[Buch1/EigTextStrukt/erbox2.tex]

5.3.2 Absatzboxen

Eine Absatzbox ist eine Box, in welcher der Text im *paragraph mode* gesetzt, d. h. wie ein Absatz behandelt wird. Um diese Boxen zu erzeugen bietet $\text{\LaTeX}$ den Befehl `\parbox` und die Umgebung `minipage`:

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

bzw.

```
\begin{minipage}[<Pos>]{<Breite>}
  <Text>
\end{minipage}
```

Bei diesen beiden Alternativen entstehen Boxen der Breite `<Breite>`, in denen Text über mehrere Zeilen und auch Absätze hinweg gesetzt werden kann. Der Text wird im Blocksatz gesetzt. Innerhalb dieses Befehls bzw. der Umgebung können nahezu alle Möglichkeiten von $\text{\LaTeX}$ genutzt werden. Der Befehl `\parbox` und die Umgebung `minipage` erzeugen keinen Leerraum, so dass man mehrere Boxen nebeneinander setzen kann. Das bedeutet, dass man gegebenenfalls den vertikalen Abstand selbst einfügen muss, am besten durch eine weitere Umgebung wie `center`, `raggedleft`, `trivlist`, u. Ä.

Die vertikale Positionierung innerhalb der Box wird durch das Argument `<Pos>` bestimmt: `b` richtet die unterste Zeile der Box an der laufenden Zeile aus, `t` die oberste Zeile der Box. Ohne Angabe des Arguments wird der Inhalt der Box vertikal zentriert.

```
8 \parbox{4.5cm}{Zeilenumbruch
9 in schmalen Absatzboxen ist
```

Zeilenumbruch in schmalen Absatzboxen ist sehr schwer.

[Buch1/EigTextStrukt/epbox.tex]

Die Umgebung `minipage` ist sehr nützlich, da sie eine vollständige kleine Seite darstellt. In ihr können Absätze, Tabellen, u. Ä. enthalten sein. Benutzt man den Befehl `\footnote` in einer `minipage`-Umgebung, so erscheint die Fußnote am Ende der Box und wird anstelle von arabischen Ziffern mit kleinen Buchstaben markiert. Wenn man `minipage`-Umgebungen allerdings schachtelt, kann die Fußnote eventuell unter der falschen Box erscheinen. Gleitobjekte können in einer `minipage`-Umgebung nicht verwendet werden. Jedoch ist es umgekehrt möglich, eine `minipage`-Umgebung in einem Gleitobjekt zu verwenden.

5.3.3 Rechteckboxen

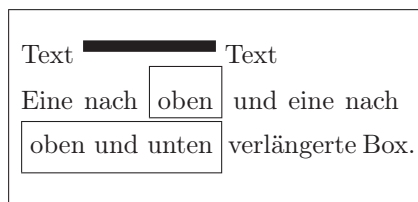
Die Rechteckboxen sind schon einmal im Kapitel Erste Schritte erwähnt worden. An dieser Stelle soll die vollständige Definition und ein weiteres Beispiel nachgereicht werden.

Eine Rechteckbox ist ein Rechteck, das mit schwarzer Farbe gefüllt ist. Im Normalfall werden damit vertikale oder horizontale Linien und Balken gezogen:

```
\rule[<Schieb>]{<Breite>}{<Höhe>}
```

Damit wird ein schwarzes Rechteck von den Argumenten `<Breite>` und `<Höhe>` erzeugt. Zusätzlich wird die Box um `<Schieb>` nach oben bzw. bei negativem Wert nach unten verschoben.

```
8 Text
9 \rule[1ex]{5em}{1ex}
10 Text
11
12 Eine nach
13 \fbox{\rule{0em}{3ex}}%
14 oben\rule{0em}{3ex}}
15 und eine nach
16 \fbox{\rule[-1ex]{0em}{3ex}}%
17 oben und unten%
18 \rule[-1ex]{0em}{3ex}}
19 verl"angerte Box.
```



[Buch1/EigTextStrukt/erbox3.tex]

Im zweiten Teil des Beispiel wird die Boxen in der Höhe variiert. Dazu wird eine Rechteckbox der Breite 0, also eine unsichtbare Linie mit einer bestimmten Höhe als Stütze, benutzt.

5.4 Maßeinheiten, Abstände und Zähler

5.4.1 Maßangaben

Maßangaben bestehen aus einer Zahl gefolgt von einer Maßeinheit. Die Zahl darf ein Vorzeichen haben (+, -); der Dezimaltrenner ist ein Punkt.

Wichtig: Auch bei der Länge 0 benötigt L^AT_EX eine Maßeinheit, beispielsweise 0pt.

```
<Zahl><Maßeinheit>
```

Gültige Maßeinheiten sind:

mm	Millimeter
cm	Zentimeter
in	Inch
pt	Point
pc	Pica
bp	Big Point
cc	Cicero
sp	Scaled Point (1pt = 65536 sp)
em	die Breite eines ‚M‘ im aktuellen Zeichensatz
ex	die Höhe eines ‚x‘ im aktuellen Zeichensatz

sp ist die Einheit, die $\text{\TeX}$ intern verwendet. Damit ergibt sich als kleinste Maßeinheit 0.0000054 mm (hieran kann man erkennen, wie genau $\text{\TeX}$ arbeitet), als größte 5.7583 m.

Die $\text{\TeX}$ -eigene Einheit *pt* ist eine Setzer-Einheit, der Pica-Punkt. Der beispielsweise von PostScript bei der Vermaßung verwendete Punkt entspricht dem *bp*.

Sehr nützlich sind die beiden relativen Maßeinheiten *em* und *ex*, denn sie passen sich der Schriftgröße an, d. h. bei Abständen, in Tabellen und Listen ist bei einem späteren Schriftwechsel nichts weiter zu tun.

Neben diesen ‚absoluten‘ Maßangaben kann man Teile bereits vorhandener Längen verwenden. Auch diese Methode hat ihre Vorteile, wenn man Schriftstücke nachträglich in der Größe ändert.

5.4.2 Abstände

Der normale Absatzabstand ist durch Standardwerte in der Dokumentklasse festgelegt. Man kann sie durch eigene Definitionen in der Präambel abändern.

Der Befehl `\baselineskip` verhält sich ein wenig anders:

```
\renewcommand{\baselinestretch}{<Faktor>}
```

Mit dem Faktor wird der Grundlinienabstand (Grundlinie engl. *baseline*) multipliziert. Ein Faktor 1.5 ergibt also eineinhalbfachen Zeilenabstand. Allerdings werden auch alle impliziten Abstände, also beispielsweise der Abstand vor und nach einer Überschrift, entsprechend vergrößert.

Um zwischen Absätzen zusätzlich senkrechten Zwischenraum zu erhalten, nutzt man die Befehle

```
\vspace{<Maßangabe>}
\vspace*{<Maßangabe>}
```

wobei bei negativer <Maßangabe> der Abstand verkürzt wird. Die *-Form erzeugt einen Zwischenraum auch dann, wenn die Anweisung nach einem Abstandsbehl oder einem Seitenwechsel erfolgt.

Abstände mit vorgegebenen Maßangaben, die mit der Dokument-Grundschrift wachsen (10pt, 11pt oder 12pt), erhält man durch die Befehle

```
\smallskip
\medskip
\bigskip
```

Hierbei ist \medskip immer doppelt so groß wie \smallskip und ebenso \bigskip immer das Doppelte von \medskip.

Anweisungen, die einen vertikalen Zwischenraum schaffen, können nur am Absatzende stehen. Ein Absatz wird beendet durch \par oder einfacher eine Leerzeile (vgl. 2.3.1, S. 28). Steht ein Abstandsbehl doch innerhalb eines Absatzes, so wird er an der nächstmöglichen Stelle gesetzt, in der Regel nach dem aktuellen Absatz.

Weitere Abstände erhält man durch folgende Anweisungen:

```
\hspace{<Maßangabe>} | \hspace*{<Maßangabe>}
```

für waagrechten Zwischenraum. Die *-Form wird immer ausgeführt, ansonsten wird der Abstand, der eine Einrückung zu Beginn eines Absatzes darstellt, ignoriert.

```
\hfill
```

ist die Kurzform des Befehls \hspace{\fill}. Er erlaubt das gleichmäßige Verteilen von Text innerhalb einer Zeile durch dehnbaren Leerraum.

`\dotfill`

ist analog zu `\hfill`, jedoch werden statt Leerraum Punkte eingefügt.

`\hrulefill`

ist analog zu `\hfill`, jedoch wird statt Leerraum eine durchgezogene Linie auf der Grundlinie eingefügt.

`\vfill`

ist die Kurzform für den Befehl `\vspace{\fill}`. Das Ergebnis ist ein vertikales Auseinanderdrücken des Texts vor und nach dem Befehl. Bei mehreren `\vfill`-Befehlen auf einer Seite werden die Absätze gleichmäßig auf einer Seite verteilt.

```

8 Adresse \hfill{}
9          Datum \
10 Nachname \dotfill{}
11 Vorname \dotfill\
12 Stra"se \hrulefill\
13 Ort \hrulefill

```

Adresse	Datum
Nachname	Vorname
Straße _____	
Ort _____	

[Buch1/EigTextStrukt/fill.tex]

5.4.3 Längenangaben

Jegliche Vermaßung in $\text{\LaTeX}$ wird über Parameter gesteuert. Das bedeutet, dass man bei Bedarf die voreingestellten Werte ändern kann. Ganz allgemein lautet die Syntax:

```
\setlength{\<Längenbefehl>}{<Maßangabe>}
```

Hiermit legt man die mit dem Befehl `<\Längenbefehl>` assoziierte Länge fest. Der frühere Wert ist damit nicht mehr verfügbar. Oft will man aber gar nicht die Länge neu setzen, sondern ein wenig vergrößern oder verkleinern. Dies kann man mit dem Befehl

```
\addtolength{\<Längenbefehl>}{<Maßangabe>}
```

der den Wert relativ zu bestehenden Wert setzt. Ist der Werte von `<Maßangabe>` negativ, dann wird vom bestehenden Wert etwas subtrahiert, sonst addiert.

```

8 \verb+\setlength+
9 \verb+{\textwidth}{16cm}+
10 setzt die Breite des Texts
11 auf 16\,cm.
12
13 \verb+\addtolength+
14 \verb+{\textwidth}{2cm}+
15 ergibt nun f"ur die Breite
16 den Wert 18\,cm.

```

```

\setlength {\textwidth}{16cm}
setzt die Breite des Texts auf 16 cm.
\addtolength {\textwidth}{2cm}
ergibt nun für die Breite den Wert
18 cm.

```

[Buch1/EigTextStrukt/laengen.tex]

5.4.4 Zähler

$\text{\LaTeX}$ verwaltet intern eine Reihe eigener Zähler. Diese haben meist dieselben Namen wie die Befehle, die sie verwenden.

Normalerweise ist der Wert eines Zählers ein ganzer, positiver Wert. Will man den Wert eines Zählers verändern, verwendet man das Kommando

```
\setcounter{\<Zähler>}{\<Wert>}
```

dies setzt den Wert des Zählers $\langle\text{Zähler}\rangle$ auf $\langle\text{Wert}\rangle$. Natürlich kann es manchmal sinnvoller sein, statt absolute Werte zu setzen, den Wert relativ zu verändern:

```
\addtocounter{\<Zähler>}{\<Wert>}
```

Der Wert des $\langle\text{Zähler}\rangle$ wird um $\langle\text{Wert}\rangle$ erhöht bzw. verringert, je nachdem, ob die Angabe positiv oder negativ ist.

```

8 \setcounter{page}{17}
9 Der Seitenzahl wird absolut auf den Wert 17
10 gesetzt. Die Anweisung \addtocounter{page}{-2}
11 \verb+\addtocounter{page}{-2}+ setzt den Seitenzahl
12 relativ zum bestehenden um 2 zurück, hier also
13 auf den Wert \thepage.

```

Der Seitenzähler wird absolut auf den Wert 17 gesetzt. Die Anweisung `\addtocounter{page}{-2}` setzt den Seitenzähler relativ zum bestehenden um 2 zurück, hier also auf den Wert 15.

Der Wert des Zählers `page` kann mit dem Befehl `\thepage` ausgegeben werden. Eine allgemeine Form für alle Zähler steht weiter unten.

Mehrfachnummern im Text werden aus verschiedenen Zählern gebildet. Wenn z. B. der Aufruf von `\subsection` die Zahl 3.7 ausgibt, setzt sich diese Zahl aus den Werten der Zähler von `\section` (3) und dem Zähler von `\subsection` (7) zusammen.

5.4.5 Ausgabe von Zählern

Mit den folgenden Anweisungen kann der aktuelle Wert eines Zählers in verschiedenen Zahlenformaten ausgegeben werden.

<code>\arabic{<Zähler>}</code>	arabische Zahlen
<code>\roman{<Zähler>}</code>	kleine römische Zahlen
<code>\Roman{<Zähler>}</code>	große römische Zahlen
<code>\alph{<Zähler>}</code>	kleine Buchstaben
	Zählerwert maximal 26
<code>\Alph{<Zähler>}</code>	große Buchstaben
	Zählerwert maximal 26

Im folgenden Beispiel hat der Zähler `chapter` den Wert 5. Dieser Wert wird dann in verschiedenen Formaten ausgegeben:

```

8 \setcounter{chapter}{5}
9 Aktuelles Kapitel:\\
10 Arabisch \arabic{chapter}\\
11 R"omisch \roman{chapter}\\
12 R"omisch \Roman{chapter}\\
13 Buchstabe \alph{chapter}\\
14 Buchstabe \Alph{chapter}

```

Aktuelles Kapitel:
 Arabisch 5
 Römisch v
 Römisch V
 Buchstabe e
 Buchstabe E

Mathematische Formeln

Ein wichtiger Anlass für die Entwicklung von $\text{\TeX}$ war der Wunsch, mathematische Formeln in hoher typografischer Qualität mit Hilfe eines Programms zu setzen. Zum damaligen Zeitpunkt gab es keine Software, die diese Anforderung erfüllen konnte, und selbst heute erreichen andere Textverarbeitungsprogramme, insbesondere beim Satz von Formeln, nicht die Qualität von $\text{\TeX}$.

Dieser hohe Anspruch wird dadurch erreicht, dass $\text{\TeX}$ einige grundlegende mathematische Begriffe ‚versteht‘ und beim Satz die für sie geltenden Konventionen berücksichtigt. $\text{\TeX}$ kennt und unterscheidet die folgenden mathematischen Kategorien:

Binäre Operatoren

wie $+$, $-$, $\cdot$, $\dots$, aber auch $\sum$, $\prod$, $\dots$ und Funktionen wie $\sin$, $\cos$, $\log$, $\arcsin$ und $\sqrt{}$.

Relationen

wie $=$, $<$, $\geq$, $\dots$

Öffnende und schließende Klammern

Die beiden Arten von Klammern werden an sich unterschieden, aber Klammersymbole wie $($ oder $)$ können, auch wenn sie im Prinzip einer Art zugerechnet werden, gemeinsam als öffnende oder schließende Klammer verwendet werden.

Variablen

Buchstaben, auch griechische, werden als Variable betrachtet. Aufeinanderfolgende Buchstaben sind das Produkt der Variablen, die aus den einzelnen Buchstaben bestehen. Alle Variable wie x oder t werden in einem speziellen Font gesetzt.

Satzzeichen

wie $,$ oder $\dots$ bei Folgen oder bei der Angabe von Vektoren.

Die Konventionen beim Satz der Zeichen, die diesen Kategorien zugeordnet sind, werden berücksichtigt. Die Zuordnung von Zeichen zu den Ka-

tegorien ist nicht festgelegt, sondern wird beispielsweise von $\text{\LaTeX}$ selbst oder von Paketen wie `amsmath` festgelegt. Grundsätzlich kann auch ein Anwender die Zuordnung vornehmen, aber $\text{\LaTeX}$ stellt dafür keine eigenen Befehle zur Verfügung.

Zeichen, die keiner dieser Kategorien zugeordnet sind, werden als *gewöhnliche Zeichen* betrachtet. Hier sind Zeichen einzuordnen wie $\forall$, ∂ , ∇ , ∞ oder $\diamond$.

6.1 Umgebungen für Formeln

$\text{\LaTeX}$ verwendet spezielle Umgebungen, um Formeln zu setzen. Im Gegensatz zum *Textmodus*, in dem Fließtext mit Absätzen, Listen und andere Umgebungen gesetzt werden, befindet man sich innerhalb der Formelumgebungen im *mathematischen Modus*, in dem die Zeichen gemäß den mathematischen Konventionen gesetzt werden. Im mathematischen Modus stehen auch spezielle Umgebungen zur Strukturierung von Formeln zur Verfügung, etwa um eine Matrix zu setzen.

Bei Formeln unterscheidet $\text{\LaTeX}$ drei Grundtypen:

1. Formeln, die innerhalb des Fließtexts auftreten.
Diese Formeln sind in der Regel relativ kurz und bestehen in vielen Fällen nur aus Variablen, die im Fließtext genannt werden. Wegen der speziellen Schriftart (engl. *font*) empfiehlt es sich aber dennoch, die Variablen als Formeln anzusehen.
2. Einzeilige abgesetzte Formeln, die zentriert in einer eigenen Zeile gesetzt werden.
3. Mehrzeilige abgesetzte Formeln.

Formeln im Fließtext (1.) werden etwas anders gesetzt als abgesetzte Formeln (2. und 3.), die in der Höhe nicht eingeschränkt sind. Im Fließtext wird eine Formel so gesetzt, dass der normale Zeilenabstand beibehalten werden kann oder – falls das nicht möglich ist – so wenig wie möglich vergrößert werden muss.

Der Begriff *Teilformel* wird im Folgenden für Teile von im mathematischen Modus geschriebenen Formeln verwendet, die auch als eigenständige Formeln gesetzt werden könnten. Insbesondere müssen zu allen in der Teilformel vorhandenen öffnenden Gruppenklammern $\{$ auch die zugehörigen schließenden $\}$ enthalten sein. Umgebungen müssen vollständig in den Teilformeln enthalten sein. Bei sich in der Größe anpassenden Klammern müssen die öffnenden Klammern ebenfalls in der Teilformel abgeschlossen werden.

Die Beispiele in diesem Kapitel beginnen und enden jeweils im Textmodus. Zunächst wird gezeigt, wie in den mathematischen Modus gewechselt und wie er beendet wird.

6.1.1 Formeln im Fließtext

In den Fließtext integrierte Formeln werden in den Zeilenumbruch mit einbezogen und bei Bedarf an geeigneten Stellen umbrochen. Für nicht abgesetzte Formeln steht die Umgebung

```
\begin{math}
  <Formel>
\end{math}
```

zur Verfügung. Die Umgebung kann nur im Textmodus verwendet werden. `<Formel>` wird im mathematischen Modus gesetzt.

Nun spielt Mathematik bei $\text{\LaTeX}$ -Dokumenten häufig eine wesentliche Rolle, und da ist es sehr lästig, jede Formel oder sogar einzelne Variablen – um sie im korrekten Schrifttyp zu setzen – mit `\begin{math}` einzuleiten und mit `\end{math}` abschließen zu müssen. Aus diesem Grund stellt $\text{\LaTeX}$ Kurzbefehle zur Verfügung, die an ihrer Stelle verwendet werden können:

```
\( Wechsel vom Textmodus in den mathematischen Modus
(statt \begin{math}).
```

```
\) Rückkehr in den Textmodus
(statt \end{math}).
```

`plain $\text{\TeX}$` verwendet übrigens zum Wechsel in den und zum Beenden des mathematischen Modus jeweils das (aktive) Zeichen `$`. Auch $\text{\LaTeX}$ akzeptiert diese noch kürzere Schreibweise, die im Folgenden allerdings nicht verwendet wird, da es sich nicht um $\text{\LaTeX}$ -Befehle handelt.

```
8 Die Variablen
9 \begin{math}x\end{math},
10 \ (y) und \ (z=x+y) sollten
11 alle im mathematischen Modus
12 geschrieben werden.
```

Die Variablen x , y und $z = x + y$ sollten alle im mathematischen Modus geschrieben werden.

[Buch1/MathForm/math.tex]

6.1.2 Einzeilige abgesetzte Formeln

Abgesetzte Formeln werden nach Voreinstellung in einer eigenen Zeile zentriert. Für einzeilige abgesetzte Formeln wird die `displaymath`-Umgebung verwendet:

```
\begin{displaymath}
  <Formel>
\end{displaymath}
```

Zeilenumbrüche sind in einzeiligen abgesetzten Formeln nicht zugelassen, wie bereits ihr Name sagt. Da anzunehmen ist, dass auch diese Formeln häufig verwendet werden, stellt $\text{\LaTeX}$ ebenfalls Abkürzungen bereit:

`\[` Wechsel in den abgesetzten mathematischen Modus
(statt `\begin{displaymath}`).

`\]` Rückkehr in den Textmodus
(statt `\end{displaymath}`).

`plainTeX` verwendet für Beginn und Ende abgesetzter Formeln übrigens jeweils `$$`. In $\text{\LaTeX}$ sollte diese Form nicht verwendet werden.

```
8 Mit den Variablen \(\mathbf{x}\) und
9 \(\mathbf{y}\) seien
10 \begin{displaymath}
11   z=f(\mathbf{x},\mathbf{y})
12 \end{displaymath}
13 und
14 \[ r=h(\mathbf{x},\mathbf{y},z) \]
15 zwei Funktionen.
```

Mit den Variablen x und y seien

$$z = f(x, y)$$

und

$$r = h(x, y, z)$$

zwei Funktionen.

[Buch1/MathForm/displaymath.tex]

Die Formel in diesem Beispiel ist entsprechend der Voreinstellung in der Zeile zentriert. Durch Angabe der Option `fleqn` in der `\documentclass`-Anweisung kann man erreichen, dass abgesetzte Formeln (ein- und mehrzeilige) statt dessen linksbündig mit einer festen Einrückung gesetzt werden. Mit `\documentclass[fleqn]{article}` wird die abgesetzte Formel wie im folgenden Beispiel ausgerichtet:

```

8 Mit den gegebenen Variablen
9 \ (x\ ) und \ (y\ ) sei
10 \ [ z=f(x,y) \ ]

```

Mit den gegebenen Variablen x und y sei

$$z = f(x, y)$$

[Buch1/MathForm/dispmath-fl.tex]

Nun möchte man sich in mathematischen Texten häufig auf frühere Formeln beziehen oder auf folgende verweisen können. In gedruckten Texten müssen Formeln dazu gekennzeichnet werden. $\LaTeX$ stellt zu diesem Zweck als Variante von `displaymath` die Umgebung

```

\begin{equation}
<Formel>
\end{equation}

```

zur Verfügung, die zusätzlich eine Formelnummer setzt. Verweise auf die vergebenen Formelnummern sind mit den üblichen Befehlen `\ref` und `\pageref` (s. 4.6) möglich. Der zugehörige `\label`-Befehl ist innerhalb der Umgebung `equation` anzugeben.

```

8 Mit den Variablen \ (x\ )
9 und \ (y\ ) seien
10 \begin{equation}
11 z=f(x,y)
12 \end{equation}
13 und
14 \begin{equation}
15 r=h(x,y,z) \label{F1}
16 \end{equation}
17 zwei Funktionen.
18 Formel~(\ref{F1}) \dots

```

Mit den Variablen x und y seien

$$z = f(x, y) \quad (1)$$

und

$$r = h(x, y, z) \quad (2)$$

zwei Funktionen. Formel (2) ...

[Buch1/MathForm/equation.tex]

6.1.3 Mehrzeilige abgesetzte Formeln

Für *mehrzeilige* abgesetzte Formeln ist die Umgebung `eqnarray` vorgesehen. Diese Umgebung entspricht einer `tabular`-Umgebung mit der Spaltenvereinbarung `{rcl}`, also mit drei Spalten, die im Unterschied zu `tabular` allerdings im mathematischen Modus gesetzt werden. Der Formelblock mit den drei Spalten wird als ganzes horizontal zentriert oder bei Angabe der `\documentclass`-Option `fleqn` mit einer Einrückung linksbündig gesetzt.

```

\begin{eqnarray}
  [<Linke Teilformel>]
    [& [<Mittlere Teilformel>]
      [& [<Rechte Teilformel>] \\\
    ... ]]]]
\end{eqnarray}

```

Die Formelzeilen werden an der mittleren Spalte ausgerichtet und einzeln nummeriert. Will man mit den üblichen Befehlen auf eine Zeile verweisen, ist vor dem die Zeile abschließenden `\\` ein `\label`-Befehl einzufügen. Die Teilformeln in den einzelnen Spalten sind beliebige Formeln, die auch unabhängig gesetzt werden könnten.

Als Variante der Umgebung ist `eqnarray*` verfügbar, bei der alle Formelnummern – wie bei `*`-Formen üblich – unterdrückt werden.

```

\begin{eqnarray*}
  ...
\end{eqnarray*}

```

Form und Inhalt der Umgebung sind genau wie bei `eqnarray`.

```

8  \begin{eqnarray}
9  f(x) & = & ax + b \\
10 h(x,y) & <> & y f(x) \\
11      & = & H(x,y) \\
12 \end{eqnarray}
13
14 \begin{eqnarray*}
15 y & = & f(x) + g(x) \\
16   & < & h(x,a) \\
17 \end{eqnarray*}

```

$$\begin{array}{rcl}
 f(x) & = & ax + b \quad (1) \\
 h(x,y) & <> & yf(x) \quad (2) \\
 & = & H(x,y) \quad (3)
 \end{array}$$

$$\begin{array}{rcl}
 y & = & f(x) + g(x) \\
 & < & h(x,a)
 \end{array}$$

[Buch1/MathForm/eqnarray.tex]

Das Beispiel zeigt, dass `eqnarray` *sämtliche* Zeilen nummeriert, während `eqnarray*` in *keiner* Formelzeile eine Formelnummer setzt.

Es kann aber sein, dass nur ausgewählte Zeilen Formelnummern erhalten sollen. Für diesen Fall ist in der Umgebung `eqnarray` der Befehl

```

\nonumber

```

vorgesehen, der bei Formelzeilen, in denen er angegeben wird, die Nummerierung unterdrückt. Es existiert hingegen kein Befehl, um Zeilen in einem `eqnarray*` zu nummerieren. Auch wenn nur eine einzige von vielen Zeilen eine Formelnummer erhalten soll, muss die Umgebung `eqnarray` verwendet und in allen nicht zu nummerierenden Zeilen der Befehl `\nonumber` angegeben werden.

```

8 \begin{eqnarray}
9 f(x) & = & ax + b \\
10 h(x,y) & = & y f(x) \\
11 & & \sqrt{x} \nonumber \\
12 & & \mbox{} + H(x,y) \\
13 \end{eqnarray}

```

$$\begin{array}{rcl}
 f(x) & = & ax + b \quad (1) \\
 h(x,y) & = & yf(x)\sqrt{x} \\
 & & + H(x,y) \quad (2)
 \end{array}$$

[Buch1/MathForm/eqnarray-n.tex]

Die `\mbox` in Zeile 12 wird verwendet, da ohne sie ein $+$ oder $-$ am Anfang einer (Teil-)Formel als Vorzeichen verstanden und daher enger an die folgende Variable gesetzt wird: `\ (+b\)` wird als $+b$ und `\ (a+b\)` als $a + b$ gesetzt. Durch diese `\mbox` wird also das $+$ zu einem Additionssymbol und mit entsprechend Abstand gesetzt.

Ein weiteres Problem tritt auf, wenn in der linken (oder auch der mittleren) Spalte der beiden mehrzeiligen mathematischen Umgebungen eine sehr breite *Teilformel* (vgl. 138) zu setzen ist und in anderen Zeilen die rechte Spalte zu breit für den verbleibenden Platz wird. Lösen lässt sich dieses Problem mit dem in beiden Umgebungen definierten Befehl

```
\lefteqn{<Teilformel>}
```

Für `<Teilformel>` gelten die gleichen Bedingungen wie für die Teilformeln in den Spalten. Nach einem `\lefteqn`-Befehl *muss* die Zeile direkt mit `\` abgeschlossen werden. Das ist erforderlich, da `\lefteqn` die Teilformel in eine Box der Breite `0pt` setzt, über die sie rechts hinausragt. Auf diese Weise wird die Spaltenbreite nicht erhöht. Teilformeln in einer folgenden Spalte würden andernfalls Teile der mit `\lefteqn` gesetzten Teilformel überschreiben.

```

8 \begin{eqnarray*}
9 \lefteqn{f(x,y)+g(x,y)} \\
10 +h(x,y)} \\
11 & = & ax+b \\
12 & & \mbox{} +2y/x+3x/y \\
13 & & \mbox{} +y(2x+c) \\
14 \end{eqnarray*}

```

$$\begin{array}{rcl}
 f(x,y)+g(x,y)+h(x,y) \\
 = & ax + b \\
 & + 2y/x + 3x/y \\
 & + y(2x + c)
 \end{array}$$

[Buch1/MathForm/eqnarray-l.tex]

Und so sähe dieses Beispiel ohne `\` aus.

```

8 \begin{eqnarray*}
9 \lefteqn{f(x,y)+g(x,y)}
10 \quad +h(x,y)}
11 & = & ax+b\backslash
12 & & \mbox{}}+2y/x+3x/y\backslash
13 & & \mbox{}}+y(2x+c)
14 \end{eqnarray*}

```

$$f(x,y)+g(x,y)+h(x,y) \\ + 2y/x + 3x/y \\ + y(2x+c)$$

[Buch1/MathForm/eqnarray-lf.tex]

6.1.4 Layout-Einstellungen für abgesetzte Formeln

Abgesetzte Formeln werden mit einem Abstand zum vorhergehenden und zum nachfolgenden Text gesetzt, und bei Angabe der Option `fleqn` werden die Formeln mit einer Einrückung linksbündig gesetzt. Diese Abstände werden durch Längen festgelegt, die der Anwender ändern kann:

Abstände bei zentrierten Formeln

`\abovedisplayskip`

Abstand einer abgesetzten Formel vom vorhergehenden Text, wenn die letzte Zeile des vorhergehenden Texts *länger* ist als der Abstand der Formel von linken Rand.

(Voreinstellung: 10pt plus 2pt minus 5pt)

`\belowdisplayskip`

Abstand einer abgesetzten Formel zum folgenden Text, wenn die letzte Zeile des vorhergehenden Texts *länger* ist als der Abstand der Formel von linken Rand.

(Voreinstellung: 10pt plus 2pt minus 5pt)

`\abovedisplayshortskip`

Abstand einer abgesetzten Formel vom vorhergehenden Text, wenn die letzte Zeile des vorhergehenden Texts *kürzer* ist als der Abstand der Formel von linken Rand.

(Voreinstellung: 0pt plus 3pt)

`\belowdisplayshortskip`

Abstand einer abgesetzten Formel zum folgenden Text, wenn die letzte Zeile des vorhergehenden Texts *kürzer* ist als der Abstand der Formel von linken Rand.

(Voreinstellung: 6pt plus 3pt minus 3pt)

Abstände bei linksbündigen Formeln mit Option `fleqn``\topsep`

Abstand einer abgesetzten Formel vom vorhergehenden und folgenden Text.

(Voreinstellung: 8pt plus 2pt minus 4pt)

`\mathindent`

Einrückung der abgesetzten Formeln. Ohne Angabe der Option `fleqn` ist diese Länge nicht definiert.

(Voreinstellung: 25pt)

Zeilenabstände in mehrzeiligen abgesetzten Formeln`\jot`

Zusätzlicher Abstand zwischen den Zeilen in `eqnarray`- und `eqnarray*`-Umgebungen.

(Voreinstellung: 3pt)

6.2 Speziell formatierte Teile von Formeln

In Formeln treten häufig Teile auf, die in besonderer Weise gesetzt werden. Dazu zählen hoch- oder tiefgestellte Teilformeln, Brüche, Wurzeln oder Matrizen. $\text{\LaTeX}$ stellt Befehle oder Umgebungen bereit, mit denen sie gesetzt werden können.

6.2.1 Hoch- und Tiefstellen von Teilformeln

Im einfachsten Fall werden Exponenten hochgestellt wie bei x^2 oder Indizes tiefgestellt wie bei x_i . Exponent und Index können aber auch komplexe Teilformeln sein, und sie können auch gleichzeitig auftreten wie bei x_i^3 . Da Hoch- und Tiefstellungen in der Mathematik sehr häufig vorkommen, verwendet $\text{\LaTeX}$ für Exponenten und Indizes jeweils ein einzelnes Befehlszeichen statt eines benannten Befehls:

`\^{<Teilformel>}`

Hochstellen einer Teilformel.

`\_{<Teilformel>}`

Tiefstellen einer Teilformel.

Alternativ kann das Hoch- und Tiefstellen auch mit den Befehlen

`\sp{<Teilformel>}` (Exponenten) und

`\sb{<Teilformel>}` (Indizes)

erfolgen. Die hoch- oder tiefgestellte Teilformel kann ihrerseits wieder hoch- oder tiefgestellte Teile enthalten. Hoch- und Tiefstellungen können also beliebig geschachtelt werden. Die Teilformel muss lediglich so aufgebaut sein, dass sie als eigenständige Formel gesetzt werden könnte.

```

8  Im Gegensatz zu Formeln
9  mit einfachen Exponenten
10 und Indizes wie
11 \ (a_{0}+a\sb{1}x+a_{2}x\sp{2})\
12 vergrößern mehrfache
13 Exponenten und Indizes wie
14 \ (x_{k_{i}}^{2})^{2^{2}}\
15 im Fließtext den
16 Zeilenabstand.
17
18 \ [x_{k_{i}}^{2})^{2^{2}}\

```

Im Gegensatz zu Formeln mit einfachen Exponenten und Indizes wie $a_0 + a_1x + a_2x^2$ vergrößern mehrfache Exponenten und Indizes wie $x_{k_i}^{2^2}$ im Fließtext den Zeilenabstand.

$$x_{k_i}^{2^2}$$

[Buch1/MathForm/expind.tex]

6.2.2 Über- und Unterklammern von Teilformeln

In mathematischen Formeln ist es nicht ungewöhnlich, dass zu Teilen einer Formel zusätzliche Angaben gemacht werden. Für diese Zusatzangaben verwendet man gerne geschweifte Klammern über oder unter dem betroffenen Teil der Formel mit den zugehörigen Angaben über oder unter der Klammer. $\text{\LaTeX}$ unterstützt diese Form von Angaben durch die beiden Befehle

`\overbrace{<Teilformel>}^{<Angaben>}`
`\underbrace{<Teilformel>}_{<Angaben>}`

```

8  \[
9  \underbrace{a, \dots, a}_{n}
10 \]
11 \[
12 \overbrace{b, \dots, b}^{m\sim
13   \text{\texttt{\scriptsize Elemente}}}
14 \]
15 \]

```

$$\overbrace{a, \dots, a}^n$$

m Elemente

$$\underbrace{b, \dots, b}_{\text{\scriptsize Elemente}}$$

[Buch1/MathForm/brace1.tex]

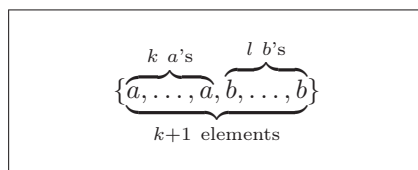
In der zweiten Formel soll nicht nur eine Variable über die Klammer gesetzt werden sondern auch Text. Mit dem Befehl `\textrm` wählt man im Mathematiksatz die Schriftart, die außerhalb des mathematischen Satzes gerade aktiv war und als Schriftausprägung geradestehend. Damit die Schriftgröße des Exponenten korrekt ist, muss man sie mit `\scriptsize` anpassen.

Eine Schachtelung der `brace`-Befehle ist möglich. Schwierig ist hingegen das Setzen überlappender Klammern, bei denen die eine rechts und die andere links über einen gemeinsamen Teil der Formel hinausragen.

```

8  \[ \{
9  \underbrace{
10 \overbrace{a, \dots, a}%
11 ^{k~a\textrm{\scriptsize
12 's}},
13 \overbrace{b, \dots, b}%
14 ^{l~b\textrm{\scriptsize
15 's}}}%
16 }_{k+1~\textrm{\scriptsize
17 elements}}
18 \] \}

```



[Buch1/MathForm/brace1a.tex]

Das Leerzeichen von dem Text *elements* wird hier durch den expliziten Abstand `~` erzeugt. Die Alternative ist wie im vorherigen Beispiel ein Leerzeichen innerhalb des `\textrm`-Arguments.

Die Ausgabe zeigt, dass die geschweifte Klammer über den *a*'s wegen der Oberlängen niedriger gesetzt wird als über den *b*'s. Damit die beiden Klammern auf der gleichen Höhe gesetzt werden, muss in das Argument des ersten `\overbrace` etwas eingefügt werden, das horizontal keinen Platz verbraucht, aber die Höhe des Buchstabens *b* hat. $\text{\LaTeX}$ bietet auch die Möglichkeit, die Höhe von gesetzten Elementen zu bestimmen und einer Länge zuzuweisen, die dann in einem `\rule`-Befehl verwendet werden kann, um eine (unsichtbare) Linie der Breite 0cm mit der richtigen Höhe zu setzen. plainTeX verfügt aber über den Befehl `\vphantom{<Element>}`, der all das von sich aus erledigt und auch in $\text{\LaTeX}$ verwendet werden kann. `<Element>` kann dabei ein Textelement oder eine Teilformel sein. Mit Hilfe dieses Befehls lässt sich das Beispiel in einfacher Weise so setzen, dass die geschweiften Klammern über den *a*'s und *b*'s auf der gleichen Höhe liegen:

```

8  \[ \{
9  \underbrace{
10   \overbrace{\vphantom{b}a,
11               \dots, a}%
12   ^{k~a\textrm{\scriptsize
13             's}}},
14   \overbrace{b, \dots, b}%
15   ^{l~b\textrm{\scriptsize
16             's}}}%
17   _{k+l~\textrm{\scriptsize
18             elements}}
19   \} \]

```

$$\overbrace{\underbrace{a, \dots, a, b, \dots, b}_{k+1 \text{ elements}}}^{k \text{ a's} \quad l \text{ b's}}$$

[Buch1/MathForm/brace1b.tex]

Für die horizontale Ausdehnung von Ausgabeelementen gibt es übrigens in plainTeX den analogen Befehl `\hphantom{<Element>}`, der bei Bedarf ebenfalls in $\text{L\AA T\TeX}$ -Dokumenten verwendet werden kann.

Besonders kompliziert ist – wie oben erwähnt – der Satz überlappender Klammern. Dies ist allerdings dennoch möglich, indem Teile der Formel, die eine der überlappenden Klammern enthalten, virtuell doppelt (übereinander) gesetzt werden. In der ersten Formel in den Zeilen 8–13 beginnen und enden die über- bzw. unterklammerten Teile der Formel an den gleichen Stellen wie bei der Formel in den Zeilen 14–19. Die schließende Klammer nach `\overbrace` in Zeile 12 wird entsprechend den Klammerregeln von $\text{L\AA T\TeX}$ als abschließende Klammer des Arguments verstanden. Entsprechend wurden ja auch die Eingabezeilen formatiert. Um nun überlappende Klammern zu erhalten, müsste diese schließende Klammer das Argument von `\underbrace` von Zeile 3 abschließen. Das ist aber nur möglich, wenn der `\overbrace`-Befehl im Argument von `\underbrace` nicht verwendet wird. Um ihn ‚nachreichen‘ zu können, muss mindestens der in der ersten Formel überklammerte Teil *doppelt* gesetzt werden, am besten einmal sichtbar und einmal unsichtbar, um ein echtes Überdrucken zu vermeiden. plainTeX verfügt über einen weiteren Befehl `\phantom{<Element>}`, der den Platz für den Satz von `<Element>` freihält, ohne es tatsächlich zu setzen. Der Befehl kann auch in $\text{L\AA T\TeX}$ für diesen Zweck verwendet werden.

Mit seiner Hilfe lassen sich nun überlappende Klammern setzen, wie das folgende Beispiel zeigt. Da es einfacher ist, den gesamten unterklammerten Teil zu überschreiben, also als `\phantom` zu setzen, als nur den in der ersten Teilformel überklammerten, wird dieser Weg gewählt. Dieser Teil wird in eine `\makebox` der Breite 0cm gesetzt, über die er nach rechts hinausragt. Nur die Klammer obendrüber wird sichtbar gesetzt. Da die `\makebox` keinen Platz beansprucht, kann anschließend die eigentliche Formel – nunmehr sichtbar – darüberschreiben und da-

bei die zweite Klammer darunter setzen, die die bereits gesetzte erste überlappt.

Das folgende Beispiel demonstriert dies in der zweiten Formel. In der ersten Formel wird noch einmal eine geschachtelte brace-Umgebung gezeigt, bei der die `\mbox` wieder zur Korrektur des ‚Operatorverstehens‘ von `TeX` eingesetzt werden muss. Nach dem Ende von `\overbrace` wird das folgende `+`-Zeichen als Vorzeichen verstanden und normalerweise enger an das folgende `l` gesetzt.

```

8  \[
9  \underbrace{i + \dots + i
10    + \overbrace{k + \dots + k}^{\scriptsize 's'}}_{\scriptsize \text{Summanden}}
11    {p \sim k \text{term} \scriptsize 's}} \mbox{}
12    + l + \dots + l}_{p+q+
13    r \sim \text{term} \scriptsize \text{Summanden}}
14  \]
15  \[
16  \makebox[0pt][l]{\(\underbrace{\phantom{i
17    + \dots + i + k
18    + \dots + k}}_{p+
19    q \sim \text{term} \scriptsize \text{Summanden}}\)}\)}%
20  i + \dots + i + \overbrace{k + \dots
21    + k + l + \dots + l}^{q+
22    r \sim \text{term} \scriptsize \text{Summanden}}
23  \]
```

$$\begin{array}{c}
 \underbrace{i + \dots + i + \overbrace{k + \dots + k}^{p \text{ } k's} + l + \dots + l}_{p+q+r \text{ Summanden}} \\
 \underbrace{i + \dots + i + \overbrace{k + \dots + k}^{q+r \text{ Summanden}} + l + \dots + l}_{p+q \text{ Summanden}}
 \end{array}$$

[Buch1/MathForm/brace2.tex]

Im Textmodus können die beiden Befehle `\overbrace` und `\underbrace` nicht verwendet werden. Im mathematischen Modus jedoch können in einer `\mbox` gesetzte Texte sehr wohl mit einer darüber oder darunter liegenden geschweiften Klammer versehen werden. Das hat allerdings in der Regel die Folge, dass die Zeile mit dem unter- oder überklammerten Text zu tief oder zu hoch wird und daher der normale Zeilenabstand nicht eingehalten werden kann. Ebenso ist zu beachten, dass der unter- oder überklammerte Text nicht umbrochen werden kann.

Die Eingabe

```
\(\underbrace{\mbox{Text mit Klammer darunter}}\)
```

setzt jetzt im Text: Text mit Klammer darunter.

6.2.3 Wurzeln

Zur Darstellung von Wurzeln wird in der Mathematik im Unterschied zu den meisten anderen Funktionen ein spezielles Zeichen verwendet. Zum Setzen einer Wurzel für eine Teilformel verwendet $\LaTeX$ den Befehl

```
\sqrt[<n-te>]{<Teilformel>}
```

mit einem optionalen Argument $\langle n-te \rangle$, das verwendet wird, wenn es sich um eine allgemeine Wurzel und nicht um die Quadratwurzel handelt. $\langle n-te \rangle$ und $\langle Teilformel \rangle$ können beliebige Teilformeln sein, die auch als eigenständige Formeln gesetzt werden können. Wird für $\langle n-te \rangle$ allerdings mehr als ein Zeichen angegeben, sieht das Ergebnis nicht mehr optimal aus. In solchen Fällen empfiehlt es sich, auf die äquivalente Exponentendarstellung der Wurzel auszuweichen. Die Höhe des Wurzelzeichens und die Länge des waagrechten Strichs werden der Größe der Teilformel angepasst. Selbstverständlich kann die Teilformel auch wieder Wurzeln enthalten.

```
8 Im Flie"stext sollten
9 nur einfache Wurzeln wie
10 \(\sqrt{3}{2}=1.259921\backslash)
11 gesetzt werden (wegen des
12 Zeilenabstands).
13
14 \[ y = \sqrt{x^{2}
15 + \sqrt{n}{x^{3} + 1/x}}
16 \]
17
18 \[ x^{1/(n^{2}+1)}
19 = \sqrt[n^{2}+1]{x}
20 \]
```

Im Fließtext sollten nur einfache Wurzeln wie $\sqrt[3]{2} = 1.259921$ gesetzt werden (wegen des Zeilenabstands).

$$y = \sqrt{x^2 + \sqrt[n]{x^3 + 1/x}}$$

$$x^{1/(n^2+1)} = \sqrt[n^2+1]{x}$$

[Buch1/MathForm/sqrt.tex]

Die letzte Formel in diesem Beispiel zeigt eine komplexere Teilformel zur Angabe einer nicht quadratischen Wurzel und die Alternative der Darstellung als Exponent. Das Paket `amsmath` erlaubt eine bessere Kontrolle

über die Positionierung des optionalen Arguments über dem Wurzelzeichen mit Hilfe des Befehl `\smash`, der in 12.1.5.10 auf Seite 411 beschrieben ist.

6.2.4 Brüche

Die letzte Formel im vorhergehenden Beispiel zeigt die Darstellung eines Bruchs mit Hilfe des Divisionszeichens. Diese Darstellung ist für Formeln im Fließtext sehr gut geeignet. Besser wird die Struktur allerdings sichtbar, wenn die Darstellung mit Hilfe eines Bruchstrichs gewählt wird. Für diese Form stellt $\text{\LaTeX}$ den Befehl

```
\frac{<Zähler>}{<Nenner>}
```

zur Verfügung. `<Zähler>` und `<Nenner>` sind auch hier wieder beliebige Teilformeln, die als eigenständige Formeln gesetzt werden könnten. Bei Formeln im Fließtext werden Zähler und Nenner in der Schriftgröße von Exponenten oder Indizes gesetzt, um nach Möglichkeit den normalen Zeilenabstand beibehalten zu können. In abgesetzten Formeln werden Zähler und Nenner hingegen in der aktuellen Schriftgröße gesetzt. Zähler und Nenner können ebenfalls Brüche sein oder Brüche enthalten. Es bleibt allerdings dem Anwender überlassen, den Hauptbruchstrich gegebenenfalls durch geeignete Maßnahmen weiter zu verlängern oder in anderer Weise die Hierarchie geschachtelter Brüche besser erkennbar zu machen. Die Ergänzungen des Pakets `amsmath` für Brüche und ähnliche Strukturen (vgl. 12.1.5.2, S. 402) sind dabei sehr hilfreich.

```

8  Im Flie"stext werden
9  Z"ahler und Nenner
10 wie
11 \(\frac{1}{2}\) oder
12 \(\frac{x}{x+y+1}\) in der
13 Schriftgr"o"se von
14 Exponenten gesetzt.
15
16 \[ 1 + x + \frac{x}{x+y+1}
17 \]
18
19 \[1 + \frac{x+3y}{
20   \frac{x^2+y^2+z^2}{
21     x+y}}
22 \]
```

Im Fließtext werden Zähler und Nenner wie $\frac{1}{2}$ oder $\frac{x}{x+y+1}$ in der Schriftgröße von Exponenten gesetzt.

$$1 + x + \frac{x}{x + y + 1}$$

$$1 + \frac{x + 3y}{\frac{x^2 + y^2 + z^2}{x + y}}$$

[Buch1/MathForm/frac.tex]

Die letzte Formel in diesem Beispiel zeigt einen Hauptbruchstrich, der nur wenig länger ist als der Bruchstrich im Nenner. In diesem Beispiel ist die Struktur aufgrund unterschiedlicher Schriftgrößen beim Zähler des Hauptbruchs einerseits und bei Zähler und Nenner des Bruchs im Nenner allerdings noch erkennbar.

6.2.5 Zeilen und Spalten

Grundlage der Darstellung von Vektoren, Matrizen und Determinanten sind tabellenartige Strukturen. Zum Setzen derartiger Strukturen steht die Umgebung `array` zur Verfügung, die praktisch identisch ist mit der Umgebung `tabular` im Textmodus (vgl. 4.2.2, S. 82). Einziger Unterschied zwischen beiden ist, dass `array` nur im mathematischen Modus verwendet werden kann und die Einträge – mit Ausnahme von `p`-Spalten – im mathematischen Modus gesetzt werden, während `tabular` nur im Textmodus erlaubt ist und alles im Textmodus setzt. Im `array` wird der Inhalt der `p`-Spalten im Textmodus gesetzt, wobei Formeln als Teil des Fließtexts wie üblich sind. Eine zur Umgebung `tabular*` äquivalente Umgebung für den mathematischen Modus gibt es nicht.

Die im mathematischen Modus verwendete Umgebung `array` hat damit die folgende Form:

```
\begin{array}[<VPos>]{<Vereinbarungen>}
  <Zeile> | \\\
  <Zeile> | ...
\end{array}
```

Das optionale Argument erlaubt wie bei `tabular` die vertikale Ausrichtung des `arrays`: bei fehlender Angabe wird es gegenüber der Grundlinie zentriert, `t` richtet die oberste und `b` die unterste Zeile auf der Grundlinie aus. `<Zeile>` steht für eine Zeile des `arrays`, bei der die Angaben für die einzelnen Spalten wie bei Tabellen durch `&` getrennt werden. Auch die `<Vereinbarungen>` sind genau wie beim `tabular` und werden daher hier nur kurz wiederholt.

Spaltenvereinbarungen

c, l, r

Eine zentriert, linksbündig oder rechtsbündig gesetzte Spalte, deren Inhalt im mathematischen Modus gesetzt wird. Einträge in diese Spalten müssen korrekte Teilformeln sein, die auch unabhängig als Formeln gesetzt werden können. Meist werden Spalten im mathematischen Modus zentriert mit `c` gesetzt.

p{<Länge>}

Eine Spalte der Breite <Länge>, die mit Randausgleich *im Textmodus* gesetzt wird.

Spaltentrennungen

- | Spaltentrennung durch einen senkrechten Strich in der Mitte des normalen Spaltenabstands.

@{<Trennung>}

Der Spaltenabstand wird durch <Trennung> ersetzt, welche im mathematischen Modus gesetzt wird. Senkrechte Linien in der Tabelle können in <Trennung> als `\vline` gesetzt werden.

Auch die sonst in der Umgebung `tabular` verfügbaren Befehle können im `array` verwendet werden:

\multicolumn{<Spzahl>}{<Vereinbarungen>}{<Eintrag>}

Eintrag über <Spzahl> Spalten entsprechend den Angaben in <Vereinbarungen>. In <Vereinbarungen> darf *nur eine* Spalte vereinbart werden. Spaltentrennungen können ohne Einschränkung verwendet werden. <Eintrag> wird im für die <Vereinbarungen> gültigen Modus gesetzt.

\vline

Eine vertikale Linie in der Höhe der Zeile. Dieser Befehl kann sowohl in Vereinbarungen im Argument von `@{...}` als auch in einem Eintrag verwendet werden.

\hline

Eine horizontale Linie über alle Spalten. Der Befehl kann nur zwischen Zeilen des `arrays` verwendet werden, also nur (a) vor der ersten Zeile, (b) nach dem Zeilenende `\\` oder (c) nach einem anderen `\hline`-Befehl.

\cline{<Startspalte>-<Endspalte>}

Eine horizontale Linie beginnend in Spalte <Startspalte> und endend in Spalte <Endspalte>. Der Befehl kann überall verwendet werden, wo auch ein `\hline`-Befehl erlaubt ist.

Von den erweiterten Möglichkeiten wird im mathematischen Modus allerdings nur selten Gebrauch gemacht. Typisch sind `arrays`, die nur zentrierte Spalten aufweisen, die lediglich durch Leerraum (engl. *white space*) getrennt sind.

```

8 Auch im Flie"stext sind
9 Strukturen wie
10 \(\begin{array}{|cc|}a & b \\\
11   b & c\end{array}\)
12 m"oglich, aber die
13 Zeilenabst"ande
14 werden gr"o"ser.
15
16 \[\begin{array}{|ccc|}
17   a^{2} & b & c \\\
18   b & \frac{a+b}{c+d} & d \\\
19   c & d & \sqrt{\frac{a}{b}}\end{array}\]
20 \end{array} \]
21
22 \[\begin{array}{|lp{2cm}|}
23   f(x) & f"ur \ (x < 0) \\\
24   g(x) & f"ur \ (x \geq 0)\end{array}\]
25 \end{array}\]

```

Auch im Fließtext sind Strukturen wie $\begin{vmatrix} a & b \\ b & c \end{vmatrix}$ möglich, aber die Zeilenabstände werden größer.

$$\begin{vmatrix} a^2 & b & c \\ b & \frac{a+b}{c+d} & d \\ c & d & \sqrt{\frac{a}{b}} \end{vmatrix}$$

$$\begin{array}{ll} f(x) & \text{für } x < 0 \\ g(x) & \text{für } x \geq 0 \end{array}$$

[Buch1/MathForm/array.tex]

Die letzte Formel des Beispiels zeigt eine p-Spalte in einem array, um Text einzufügen. Der Text in der zweiten Spalte enthält sogar wieder eine kurze Formel.

Bis auf eine Ausnahme sind Layout-Einstellungen für beide Umgebungen `array` und `tabular` gültig. Die Ausnahme ist der Spaltenabstand, der für `array` und `tabular` getrennt eingestellt wird. Die folgenden Stilparameter werden von `array` berücksichtigt:

`\arraycolsep`

Halber Spaltenabstand in einem array (beim `tabular` wird statt dessen `\tabcolsep` verwendet).
(Voreinstellung: 5pt)

`\arrayrulewidth`

Strichstärke der horizontalen und vertikalen Linien.
(Voreinstellung: 0.4pt)

`\doublerulesep`

Abstand zweier direkt aufeinander folgenden horizontalen oder vertikalen Linien.
(Voreinstellung: 2pt)

`\arraystretch`

Ein Befehl zur Kontrolle des Zeilenabstands, der als Faktor des voreingestellten Zeilenabstands verwendet wird.
(Voreinstellung: 1)

Weitere Anpassungen können mit Hilfe des Pakets `array` (siehe 9.2) vorgenommen werden.

6.3 Symbole

In der Mathematik wird eine Vielzahl von Symbolen verwendet. Dazu zählen beispielsweise griechische und kalligrafische Buchstaben, spezielle Operatoren, die unterschiedlichsten Pfeile und geometrische Formen. Die am häufigsten verwendeten Symbole stellt $\text{\LaTeX}$ zur Verfügung. Einige ergänzende Symbole definiert das Paket `latexsym`, das zur Grundausstattung von $\text{\LaTeX}$ gehört. Ungefähr 250 weitere Symbole und Varianten vereinbart das Paket `amssymb` (siehe 13.3.2) der $\mathcal{AMS}$ (American_Mathematical_Society). Es ist daher sehr selten, dass beim Satz von Formeln das richtige Zeichen fehlt.

6.3.1 Griechische Buchstaben

In mathematischen Formeln werden häufig griechische Buchstaben verwendet, beispielsweise für Winkel. Die kleinen griechischen Buchstaben unterscheiden sich alle von etwa im Deutschen oder Englischen verwendeten und müssen daher vollständig mit speziellen Befehlen gesetzt werden. Ein Teil der griechischen Großbuchstaben entspricht den im lateinischen gebräuchlichen, wenn auch gelegentlich mit anderer Bedeutung. Bei den Großbuchstaben müssen daher nur die fehlenden durch Befehle gesetzt werden. Tabelle 6.1 zeigt die griechischen Buchstaben zusammen mit den Befehlen, mit denen sie gesetzt werden. Die griechischen Buchstaben stehen nur im mathematischen Modus zur Verfügung, falls nicht die Spracheinstellungen für Griechisch aktiv sind. Die $\mathcal{AMS}$ stellt zusätzlich kursive Varianten der griechischen Großbuchstaben zur Verfügung (siehe Tabelle 12.1).

6.3.2 Kalligrafische Großbuchstaben

$\text{\LaTeX}$ verfügt über 26 kalligrafische Großbuchstaben, die im mathematischen Modus verwendet werden können. Der Befehl zum Setzen der kalligrafischen Buchstaben lautet

```
\mathcal{<Großbuchstabe (n)>}
```

Tabelle 6.1: Griechische Buchstaben

Kleinbuchstaben			
α <code>\alpha</code>	θ <code>\theta</code>	o <code>o</code>	τ <code>\tau</code>
β <code>\beta</code>	ϑ <code>\vartheta</code>	π <code>\pi</code>	υ <code>\upsilon</code>
γ <code>\gamma</code>	ι <code>\iota</code>	ϖ <code>\varpi</code>	ϕ <code>\phi</code>
δ <code>\delta</code>	κ <code>\kappa</code>	ρ <code>\rho</code>	φ <code>\varphi</code>
ϵ <code>\epsilon</code>	λ <code>\lambda</code>	ϱ <code>\varrho</code>	χ <code>\chi</code>
ε <code>\varepsilon</code>	μ <code>\mu</code>	σ <code>\sigma</code>	ψ <code>\psi</code>
ζ <code>\zeta</code>	ν <code>\nu</code>	ς <code>\varsigma</code>	ω <code>\omega</code>
η <code>\eta</code>	ξ <code>\xi</code>		
Großbuchstaben			
Γ <code>\Gamma</code>	Λ <code>\Lambda</code>	Σ <code>\Sigma</code>	Ψ <code>\Psi</code>
Δ <code>\Delta</code>	Ξ <code>\Xi</code>	Υ <code>\Upsilon</code>	Ω <code>\Omega</code>
Θ <code>\Theta</code>	Π <code>\Pi</code>	Φ <code>\Phi</code>	

und ist auf den mathematischen Modus beschränkt. Die alte Form dieses Befehls `\cal{...}` kann ebenfalls noch verwendet werden, auch wenn davon abzuraten ist. Das folgende Beispiel zeigt die kalligrafischen Buchstaben.

```
8 \(\mathcal{ABCDEFGHJKLMN}\)
9 \(\mathcal{NOPQRSTUVWXYZ}\)
```

ABCDEFGHJKLMN
NOPQRSTUVWXYZ

[Buch1/MathForm/cal.tex]

6.3.3 Schriften im Formeln

Im Unterschied zu den kalligrafischen Großbuchstaben stehen bei den hier behandelten auch Kleinbuchstaben, Ziffern und Satzzeichen zur Verfügung. Umlaute und Eszett fehlen allerdings in diesen Schriften.

```
\mathnormal{<Zeichen>} (AaxXzZ) – 0123456789!
```

`\mathnormal` ist der Standardfont im mathematischen Modus.

Die Zeichen stimmen mit Ausnahme der Ziffern, die von `\mathnormal` als ‚oldstyle‘ Ziffern gesetzt werden, mit denen von `\mathtt` gesetzten überein.

```
\mathrm{<Zeichen>} (AaxXzZ) – 0123456789!
```

Analog zum Standardfont im Textmodus (`\textrm`).

```
\mathbf{<Zeichen>} (AaxXzZ) – 0123456789!
```

Fetter Zeichensatz analog zum Textfont (`\textbf`).

`\mathsf{<Zeichen>}` (AaxXzZ) – 0123456789!

Serifenloser Zeichensatz (Textfont `\textsf`).

`\mathit{<Zeichen>}` (AaxXzZ) – 0123456789!

Kursiver Zeichensatz (im Text `\textit`).

`\mathtt{<Zeichen>}` (AaxXzZ) – 0123456789!

Schreibmaschinenschrift, analog zum Textmodus (`\texttt`).

`\math`... setzt die als Argument angegebenen Zeichen nicht als Text, sondern wie andere Zeichen im mathematischen Modus. Daher werden Leerzeichen ignoriert. Weiter ist zu beachten, dass im mathematischen Modus Schriften und nicht Schriftattribute, wie im Textmodus ausgewählt werden. Eine Kombination der Stileigenschaften ist daher nicht möglich.

6.3.4 Binäre Operatoren

Neben + und – kennt $\text{\LaTeX}$ eine Reihe weiterer binärer Operatoren. Vier weitere Operatoren definiert das Grundpaket `latexsym`.

Tabelle 6.2 zeigt die in $\text{\LaTeX}$ vorgesehenen Operatoren zusammen mit den Befehlen, um sie zu setzen. Zusätzlich sind die Operatoren aus dem Grundpaket `latexsym` in einem eigenen Teil der Tabelle aufgeführt.

Die in der späteren Tabelle 13.5 auf Seite 489 aufgeführten Operatoren stellt die $\mathcal{A}\mathcal{M}\mathcal{S}$ im Paket `amssymb` zur Verfügung. Das in 12.2.3 auf Seite 423 behandelte Paket `amsofn` vereinbart einige weitere Operatoren und erlaubt darüber hinaus die Definition eigener Operatoren. Die Definition eigener Operatoren durch den Benutzer wird aber nur in sehr seltenen Fällen notwendig sein.

Tabelle 6.2: Benannte binäre Operatoren

$\pm$ <code>\pm</code>	$\oplus$ <code>\oplus</code>	$\cap$ <code>\cap</code>	$\triangleup$ <code>\bigtriangleup</code>
$\mp$ <code>\mp</code>	$\ominus$ <code>\ominus</code>	$\cup$ <code>\cup</code>	$\triangledown$ <code>\bigtriangledown</code>
$\times$ <code>\times</code>	$\otimes$ <code>\otimes</code>	$\uplus$ <code>\uplus</code>	$\triangleleft$ <code>\triangleleft</code>
$\div$ <code>\div</code>	$\oslash$ <code>\oslash</code>	$\sqcap$ <code>\sqcap</code>	$\triangleright$ <code>\triangleright</code>
$\cdot$ <code>\cdot</code>	$\odot$ <code>\odot</code>	$\sqcup$ <code>\sqcup</code>	$\bullet$ <code>\bullet</code>
$\star$ <code>\star</code>	$\diamond$ <code>\diamond</code>	$\setminus$ <code>\setminus</code>	$\bigcirc$ <code>\bigcirc</code>
$\ast$ <code>\ast</code>	$\dagger$ <code>\dagger</code>	$\vee$ <code>\vee</code>	$\amalg$ <code>\amalg</code>
$\circ$ <code>\circ</code>	$\ddagger$ <code>\ddagger</code>	$\wedge$ <code>\wedge</code>	$\wr$ <code>\wr</code>

Zusätzliche binäre Operatoren im Grundpaket `latexsym`

$\trianglelefteq$ <code>\trianglelefteq</code>	$\trianglerighteq$ <code>\trianglerighteq</code>	$\trianglelefteqslant$ <code>\trianglelefteqslant</code>	$\trianglerighteqslant$ <code>\trianglerighteqslant</code>
--	--	--	--

Neben den eigentlichen $\text{\LaTeX}$ -Operatoren kann man in $\text{\LaTeX}$ auch Operatoren von plainTeX verwenden. Das ist jedoch nur in wenigen Fällen sinnvoll. Zwei dieser Operatoren von plainTeX können allerdings ansonsten nur im Paket `amsmath` verfügbare Operatoren ersetzen. Damit kann auf das Laden von `amsmath` verzichtet werden kann, wenn andere Erweiterungen aus dem Paket nicht benötigt werden. Es handelt sich um die beiden Operatoren

`\choose` zum Setzen von Binomialkoeffizienten und

`\atop` zum Setzen von zwei Teilformeln übereinander ähnlich wie mit `\stackrel`, nur dass die beiden Teilformeln in der gleichen Größe gesetzt werden.

Beide Operatoren sind einander sehr ähnlich. Sie unterscheiden sich lediglich durch die zusätzlichen runden Klammern bei den Binomialkoeffizienten. Das folgende Beispiel zeigt die beiden Operatoren.

```

8 \[
9 {n+k \choose k} \qquad
10 {a+b+c \atop x+y}
11 \]
```

$$\begin{array}{cc} \binom{n+k}{k} & a+b+c \\ & x+y \end{array}$$

[Buch1/MathForm/plainops.tex]

Der `\atop`-Operator ist auch für zweizeilige Angaben unterhalb großer Operatoren wie beispielsweise `\bigtriangleup` gut geeignet.

6.3.5 Relationen

Die Relationen sind eine recht große Gruppe unter den Symbolen. Hierher gehören neben $=$, $\leq$, $\dots$ beispielsweise auch Beziehungen zwischen Mengen, Ähnlichkeitsrelationen und verschiedenartige Pfeile. Die Pfeile sind als vergleichsweise große Gruppe in einer eigenen Tabelle aufgeführt, obwohl sie sich in ihrer Eigenschaft als Relationen nicht von den anderen unterscheiden.

Tabelle 6.3 gibt eine Übersicht über die Relationen mit Ausnahme der Pfeile, die in Tabelle 6.4 zusammengefasst sind. Das Grundpaket `latexsym` definiert vier weitere Relationen, die in Tabelle 6.3 ebenfalls aufgeführt sind. Viele weitere in den Tabellen 13.3 und 13.4 auf den Seiten 488 und 489 gezeigte Relationen stellt die $\mathcal{AMS}$ in ihrem Paket `amssymb` zur Verfügung.

Tabelle 6.3: Relationen

$<$	$<$	$>$	$>$	$=$	$=$	$\doteq$	<code>\doteq</code>
$\leq$	<code>\leq</code>	$\geq$	<code>\geq</code>	$\neq$	<code>\neq</code>	$\propto$	<code>\propto</code>
$\prec$	<code>\prec</code>	$\succ$	<code>\succ</code>	$\sim$	<code>\sim</code>	$\approx$	<code>\approx</code>
$\preceq$	<code>\preceq</code>	$\succeq$	<code>\succeq</code>	$\simeq$	<code>\simeq</code>	$\cong$	<code>\cong</code>
$\ll$	<code>\ll</code>	$\gg$	<code>\gg</code>	$\smile$	<code>\smile</code>	$\frown$	<code>\frown</code>
$\subset$	<code>\subset</code>	$\supset$	<code>\supset</code>	$\asymp$	<code>\asymp</code>	$\equiv$	<code>\equiv</code>
$\subseteq$	<code>\subseteq</code>	$\supseteq$	<code>\supseteq</code>	$\mid$	<code>\mid</code>	$\parallel$	<code>\parallel</code>
$\sqsubseteq$	<code>\sqsubseteq</code>	$\sqsupseteq$	<code>\sqsupseteq</code>	$\models$	<code>\models</code>	$\perp$	<code>\perp</code>
$\in$	<code>\in</code>	$\ni$	<code>\ni</code>	$\vdash$	<code>\vdash</code>	$\dashv$	<code>\dashv</code>
$\bowtie$	<code>\bowtie</code>						

Zusätzliche Relationen im Grundpaket latexsym

$\sqsubset$	<code>\sqsubset</code>	$\sqsupset$	<code>\sqsupset</code>	$\Join$	<code>\Join</code>	$\leadsto$	<code>\leadsto</code>
-------------	------------------------	-------------	------------------------	---------	--------------------	------------	-----------------------

Die von $\text{\LaTeX}$ zur Verfügung gestellten Pfeile – auch sie gehören zu den Relationen – zeigt Tabelle 6.4. Weitere Pfeile und negierte Pfeile enthalten die Fonts der $\mathcal{A}\mathcal{M}\mathcal{S}$ (siehe Tabelle 13.2)

Tabelle 6.4: Pfeile (spezielle Relationen)

$\leftarrow$	<code>\leftarrow</code>	$\Leftarrow$	<code>\Leftarrow</code>
$\rightarrow$	<code>\rightarrow</code>	$\Rightarrow$	<code>\Rightarrow</code>
$\leftrightarrow$	<code>\leftrightarrow</code>	$\Leftrightarrow$	<code>\Leftrightarrow</code>
$\longleftarrow$	<code>\longleftarrow</code>	$\Longleftarrow$	<code>\Longleftarrow</code>
$\longrightarrow$	<code>\longrightarrow</code>	$\Longrightarrow$	<code>\Longrightarrow</code>
$\longleftrightarrow$	<code>\longleftrightarrow</code>	$\Longleftrightarrow$	<code>\Longleftrightarrow</code>
$\uparrow$	<code>\uparrow</code>	$\Uparrow$	<code>\Uparrow</code>
$\downarrow$	<code>\downarrow</code>	$\Downarrow$	<code>\Downarrow</code>
$\updownarrow$	<code>\updownarrow</code>	$\Updownarrow$	<code>\Updownarrow</code>
$\nwarrow$	<code>\nwarrow</code>	$\swarrow$	<code>\swarrow</code>
$\nearrow$	<code>\nearrow</code>	$\searrow$	<code>\searrow</code>
$\hookleftarrow$	<code>\hookleftarrow</code>	$\hookrightarrow$	<code>\hookrightarrow</code>
$\longmapsto$	<code>\longmapsto</code>	$\mapsto$	<code>\mapsto</code>
$\leftharpoonup$	<code>\leftharpoonup</code>	$\leftharpoonown$	<code>\leftharpoonown</code>
$\rightharpoonup$	<code>\rightharpoonup</code>	$\rightharpoonown$	<code>\rightharpoonown</code>
$\rightleftharpoons$	<code>\rightleftharpoons</code>		

In der Mathematik ist es üblich Relationen durchzustreichen, wenn sie *nicht* gelten. $\text{\LaTeX}$ verwendet zum Negieren von Relationen den Befehl

```
\not
```

```

8 \[
9 a\not=b \quad
10 y\notin X
11 \]
```

$$a \neq b \quad y \notin X$$

[Buch1/MathForm/not.tex]

Der `\not`-Befehl kann auch mit anderen Symbolen verwendet werden, aber die Position des Schrägstrichs ist speziell auf die Relationen in Tabelle 6.3 abgestimmt, die die Breite von $=$, $\leq$ oder $\in$ haben. Bei schmalen oder breiteren Symbolen sitzt der Schrägstrich nicht in der Mitte des Symbols.

Werden Pfeile als Symbole für Abbildungen eingesetzt, schreibt man in der Mathematik häufig die Bezeichnung für die Abbildung über den Pfeil. Auch bei anderen Symbolen setzt man gelegentlich zusätzliche Angaben über das Symbol. $\LaTeX$ stellt den Befehl

```
\stackrel{<Obere Teilformel>}{<Untere Teilformel>}
```

zur Verfügung, der das Argument *<Obere Teilformel>* in der Schriftgröße eines Exponenten *über* das zweite Argument *<Untere Teilformel>* (in der normalen Schriftgröße) setzt. Bereits der Name des Befehls weist darauf hin, dass das aus den beiden Argumenten zusammengesetzte Symbol wie eine Relation behandelt wird. Als Argumente können zwar beliebige Teilformeln angegeben werden, aber eigentlich wird angenommen, dass *<Untere Teilformel>* selbst eine Relation oder vielleicht eine Funktionsbezeichnung ist und *<Obere Teilformel>* eine sehr einfache Funktion. Die Angabe komplexerer Funktionen ist zwar möglich, aber nicht zu empfehlen.

Das folgende Beispiel zeigt Anwendungen der beiden Befehle `\not` und `\stackrel`.

```

8  \[
9  a
10 \stackrel{f_x}{\longrightarrow} b
11 b
12 \stackrel{g}{\longrightarrow} c
13 c
14 \]
15 \[ f(x) \not\leq g(x) \]
16 \[ x_1 \not\parallel b,
17   u \not\longleftarrow b \]
18 \[ x_1 \hspace{-.5mm}\not
19   \hspace{.5mm}\parallel b,
20   u\hspace{.75em}
21   \not\hspace{-.75em}
22   \longleftarrow b
23 \]

```

$$a \xrightarrow{f_x} b \xrightarrow{g} c$$

$$f(x) \not\leq g(x)$$

$$x_1 \not\parallel b, u \not\longleftarrow b$$

$$x_1 \not\parallel b, u \not\longleftarrow b$$

[Buch1/MathForm/relationen.tex]

Die Formeln in den Zeilen 16 und 17 zeigt die Anwendung von `\not` auf ein zu schmales und ein zu breites Zeichen, bei denen die automatische Positionierung durch $\text{\LaTeX}$ nicht erfolgreich ist. In den Zeilen 18–22 ist die gleiche Formel mit manueller Korrektur der Abstände gesetzt, um zu zeigen, dass sich auf diese Weise `\not` auch auf ‚nicht passende‘ Zeichen anwenden lässt. Es genügt nicht nur den Schrägstrich richtig zu positionieren. Die manuelle Korrektur durch den negative `\hspace` muss anschließend wieder in derselben Breite wieder eingefügt werden, damit der Abstand auch für das nachfolgende Zeichen stimmt. $\text{\LaTeX}$ ist in der Lage die Breite für ein Zeichen zu messen und so die Korrektur für das `\not` selbst zu kalkulieren, das benötigte Paket heißt `calc` (s. 9.4, S. 246). Allerdings lohnt sich dieser Aufwand in der Regel nicht.

6.3.6 Große Operatoren

Neben den normalen Operatoren wie `+` oder `·` verwendet $\text{\LaTeX}$ auch sogenannte *große Operatoren*. Sie werden im Unterschied zu den normalen Operatoren in abgesetzten Formeln größer gesetzt als in Formeln im Fließtext. Tabelle 6.5 zeigt die großen Operatoren.

Das folgende Beispiel zeigt die unterschiedliche Größe in Formeln im Fließtext und in abgesetzten Formeln. Ein weiterer Unterschied ist, dass im Fließtext die Angaben zum Produkt in Form von Index und Exponent gesetzt werden, bei der abgesetzten Formel hingegen unter und über dem Produktzeichen. Andererseits wird beim Integral in beiden Fällen die Form als Index und Exponent gewählt.

Tabelle 6.5: Operatoren in variabler Größe

Σ	$\sum$	<code>\sum</code>	$\cap$	$\bigcap$	<code>\bigcap</code>	$\odot$	$\bigodot$	<code>\bigodot</code>
Π	$\prod$	<code>\prod</code>	$\cup$	$\bigcup$	<code>\bigcup</code>	$\otimes$	$\bigotimes$	<code>\bigotimes</code>
$\coprod$	$\coprod$	<code>\coprod</code>	$\sqcup$	$\bigsqcup$	<code>\bigsqcup</code>	$\oplus$	$\bigoplus$	<code>\bigoplus</code>
$\int$	$\int$	<code>\int</code>	$\vee$	$\bigvee$	<code>\bigvee</code>	$\uplus$	$\biguplus$	<code>\biguplus</code>
$\oint$	$\oint$	<code>\oint</code>	$\wedge$	$\bigwedge$	<code>\bigwedge</code>			

```

8 \begin{center}
9 \ (
10 \prod_{i=1}^n c_i/(1+c_i^2)
11 \ )\ [2mm]
12 \ ( \int_0^\infty e^{-x^2} \ )
13 \end{center}
14
15 \ [ \prod_{i=1}^n
16 \ \frac{c_i}{1+c_i^2} \ ]
17 \ [ \int_0^\infty e^{-x^2} \ ]

```

$$\prod_{i=1}^n c_i/(1+c_i^2)$$

$$\int_0^\infty e^{-x^2}$$

$$\prod_{i=1}^n \frac{c_i}{1+c_i^2}$$

$$\int_0^\infty e^{-x^2}$$

[Buch1/MathForm/bigop.tex]

6.3.7 Funktionen

Funktionsbezeichnungen werden üblicherweise im Unterschied zu den Variablennamen in der gleichen Schrift wie der Text gesetzt. $\text{\LaTeX}$ stellt zwar Befehle zur Verfügung, mit denen Funktionsbezeichnungen in dieser Form korrekt gesetzt werden können, aber zur Vereinfachung sind für häufiger verwendete Funktionen Befehle vorgesehen, um die Funktionsbezeichnung zu setzen. Technisch gesehen sind diese Funktionen für $\text{\LaTeX}$ Operatoren. Tabelle 6.6 listet die vordefinierten Funktionen auf.

6.3.8 Klammern

Ein wichtiges Element in mathematischen Formeln sind Klammern. Auch $\text{\LaTeX}$ kennt eine Reihe von Klammern unterschiedlicher Form. $\text{\LaTeX}$ kennt die in Tabelle 6.7 aufgeführten Klammersymbole. $\text{\LaTeX}$ unterscheidet dabei nicht zwischen öffnenden und schließenden Klammern, so dass

Tabelle 6.6: Vordefinierte Funktionen

$\sin$	<code>\sin</code>	$\sinh$	<code>\sinh</code>	$\arcsin$	<code>\arcsin</code>	$\exp$	<code>\exp</code>
$\cos$	<code>\cos</code>	$\cosh$	<code>\cosh</code>	$\arccos$	<code>\arccos</code>	$\ln$	<code>\ln</code>
$\tan$	<code>\tan</code>	$\tanh$	<code>\tanh</code>	$\arctan$	<code>\arctan</code>	$\log$	<code>\log</code>
$\cot$	<code>\cot</code>	$\coth$	<code>\coth</code>	$\arg$	<code>\arg</code>	$\lg$	<code>\lg</code>
$\sec$	<code>\sec</code>	$\lim$	<code>\lim</code>	$\dim$	<code>\dim</code>	$\det$	<code>\det</code>
$\csc$	<code>\csc</code>	$\liminf$	<code>\liminf</code>	$\inf$	<code>\inf</code>	$\min$	<code>\min</code>
$\deg$	<code>\deg</code>	$\limsup$	<code>\limsup</code>	$\sup$	<code>\sup</code>	$\max$	<code>\max</code>
$\gcd$	<code>\gcd</code>	$\hom$	<code>\hom</code>	$\ker$	<code>\ker</code>	$\Pr$	<code>\Pr</code>

] als öffnende ebenso wie als schließende Klammern verwendet werden kann, wie es bei halboffenen Intervallen wie $]a, b]$ gebräuchlich ist. Die in Tabelle 13.6 wiedergegebenen weiteren Klammersymbole stellt die AMS in ihrem Paket `amfonts` zur Verfügung.

Tabelle 6.7: Klammersymbole (vgl. auch `amfonts` Tab. 13.6, S. 490)

$($	<code>(</code>	$)$	<code>)</code>	$\lfloor$	<code>\lfloor</code>	$\rfloor$	<code>\rfloor</code>
$[$	<code>[</code>	$]$	<code>]</code>	$\lceil$	<code>\lceil</code>	$\rceil$	<code>\rceil</code>
$\{$	<code>\{</code>	$\}$	<code>\}</code>	$\langle$	<code>\langle</code>	$\rangle$	<code>\rangle</code>
$/$	<code>/</code>	$\backslash$	<code>\backslash</code>	$\updownarrow$	<code>\updownarrow</code>	$\Updownarrow$	<code>\Updownarrow</code>
$\uparrow$	<code>\uparrow</code>	$\Uparrow$	<code>\Uparrow</code>	$\downarrow$	<code>\downarrow</code>	$\Downarrow$	<code>\Downarrow</code>
$ $	<code> </code>	$\ $	<code>\ </code>				

Die Grundgröße der Klammern ist allerdings in vielen Fällen nicht ausreichend, beispielsweise im Fall der runden Klammern um eine Matrix. Aus diesem Grund bietet $\text{\LaTeX}$ zwei Methoden, größere Klammern zu setzen: Zum einen gibt es Befehle, um Klammern in vier verschiedenen Größen zu setzen. Andererseits stehen Befehle zur Verfügung, um Klammern zu setzen, die sich in ihrer Größe an die von ihnen eingeschlossene Teilformel anpassen. In beiden Fällen müssen entsprechende Befehle bei *beiden* Klammern verwendet werden.

Gelegentlich verwendet man in der Mathematik allerdings nur eine einzelne (vergrößerte) Klammer, beispielsweise bei Fallunterscheidungen. Bei den großen Klammern darf aber – wie oben gesagt – weder die öffnende noch die schließende Klammer fehlen. Aus diesem Grund kennt $\text{\LaTeX}$ ein weiteres Klammersymbol, und zwar die *unsichtbare Klammer*:

- Unsichtbare Klammer zum Gebrauch mit den Befehlen zum Vergrößern von Klammern

Die unsichtbare Klammer tritt also bei vergrößerten Klammern an die Stelle einer fehlenden. An anderen Stellen als nach den Befehlen zum

Vergrößern von Klammern hat der Punkt seine übliche Bedeutung und wird als Zeichen gesetzt.

Die folgenden Befehle dienen zum Setzen der Klammersymbole aus Tabelle 6.7 in vier Vergrößerungsstufen. `<KS>` steht dabei für eines der Klammersymbole aus der Tabelle oder die unsichtbare Klammer.

`\bigl<KS>` ... `\bigr<KS>`

Erste Vergrößerungsstufe

`\Bigl<KS>` ... `\Bigr<KS>`

Zweite Vergrößerungsstufe

`\biggl<KS>` ... `\biggr<KS>`

Dritte Vergrößerungsstufe

`\Biggl<KS>` ... `\Biggr<KS>`

Vierte und größte Vergrößerungsstufe

In einer Formel müssen *stets beide* aufgeführten Befehle verwendet werden. Die Größe der Klammern – die unverändert festliegt – zeigt das abschließende Beispiel.

Klammern, die sich in ihrer Größe an die eingeschlossene Teilformel anpassen, lassen sich mit

`\left<KS>` ... `\right<KS>`

setzen. `<KS>` steht auch hier für eines der Klammersymbole oder die unsichtbare Klammer. Wieder müssen `\left` und `\right` *beide* in einer Formel verwendet werden. Mit diesen Befehlen werden Klammern nur so weit vergrößert, dass sie nicht kleiner als die von ihnen eingeschlossene Teilformel sind. Sie sind nur dann größer als innere Klammern, wenn an Teile der Formel höher oder tiefer reichen als die inneren Klammern. Das folgende Beispiel zeigt auch die Anwendung dieser Befehle.

```

8  \[\left(\left(\left(
9    \left((x+y)\right)^2
10   \right)\right)\right)\]
11  \[\Bigl(\biggl(\Bigl(
12    \bigl((x+y)\bigr)^2
13    \Bigr)\biggr)\Bigr)\]
14  \[\Biggl(\biggl(\Bigl(
15    \bigl((x+y)\bigr)
16    \Bigr)\biggr)\Bigr)\]
17  \[\left(\prod_{i=1}^n
18    \frac{c_i}{1+c_i^2}
19    \right) \quad \quad
20    \left[\int\limits_0^\infty
21    e^{-x^2}\right]\]
22  \[h(x)=\left\{
23    \begin{array}{ll}
24      0, & x \leq 0 \\
25      x^2, & x > 0
26    \end{array}\right.\]

```

$$\begin{array}{c}
 (((((x+y)^2))) \\
 \left(\left(\left((x+y)^2 \right) \right) \right) \\
 \left(\left(\left((x+y) \right) \right) \right) \\
 \left(\left(\left((x+y) \right) \right) \right) \\
 \left(\prod_{i=1}^n \frac{c_i}{1+c_i^2} \right) \quad \left[\int_0^\infty e^{-x^2} \right] \\
 h(x) = \begin{cases} 0, & x \leq 0 \\ x^2, & x > 0 \end{cases}
 \end{array}$$

[Buch1/MathForm/vbraces.tex]

An der ersten Formel in den Zeilen 5–7 ist zu sehen, dass die Klammern nur vergrößert werden, wenn – wie in Zeile 6 durch den Exponent – die Höhe der eingeschlossenen Teilformel zunimmt. Ein Vergleich der beiden Formeln in den Zeilen 8–10 und 11–13 zeigt, dass bei den hier verwendeten Befehlen die Höhe der eingeschlossenen Teilformel keinen Einfluss auf die Größe der Klammer hat. An der Formel in den Zeilen 14–18 kann man die Flexibilität von `\left` und `\right` erkennen. Die letzte Formel in den Zeilen 19–23 ist ein Beispiel für die unsichtbare Klammer.

6.3.9 Verschiedene Arten von Auslassungspunkten

Während im Textmodus für Folgepunkte nur der Befehl `\dots` zur Verfügung steht, kennt der mathematische Modus mehrere Arten von Auslassungspunkten:

`\ldots` | `\dots`

Folgepunkte auf der Grundlinie; die Befehle können auch im Textmodus verwendet werden.

`\cdots` Folgepunkte zwischen Operatoren wie $+$, $-$ oder Relationen wie $=$. Diese Folgepunkte liegen auf der Höhe des Operators $=$, also auf der Mittellinie.

`\vdots` Drei vertikale Folgepunkte, die typischerweise in array-Umgebungen verwendet werden, wenn Zeilen ausgelassen werden; der Befehl kann auch im Textmodus verwendet werden.

`\ddots` Drei diagonale Folgepunkte von links oben nach rechts unten. Diagonale Folgepunkte werden ebenfalls vor allem in arrays verwendet, um Fortsetzungen in Diagonalen anzudeuten.

Das folgende Beispiel zeigt die Verwendung der Folgepunkte.

```

8  Eine Folge
9  \((a_{1}, a_{2}, \ldots)\),
10 ein Vektor
11 \((x_{1}, \ldots, x_{n})\)
12 und eine Summe
13 \((a_{1} + \cdots + a_{n})\)
14 verwenden hier Folgepunkte
15 \dots{}
16
17 \[
18 \begin{array}{cccc}
19 d & a_{1,2} & \dots & a_{1,i} \\
20 a_{2,1} & d & & a_{2,i} \\
21 \vdots & & \ddots & \vdots \\
22 a_{j,1} & \dots & a_{j,i-1} & d
23 \vdots & & \ddots & \vdots \\
24 \vdots & & \ddots & \vdots
25 \end{array}
26 \]
27
28 Und im Text \dots{} \vdots{} \ddots

```

Eine Folge $a_1, a_2, \dots$, ein Vektor $(x_1, \dots, x_n)$ und eine Summe $a_1 + \dots + a_n$ verwenden hier Folgepunkte ...

$$\begin{array}{cccc}
 d & a_{1,2} & \dots & a_{1,i} \\
 a_{2,1} & d & & a_{2,i} \\
 \vdots & & \ddots & \vdots \\
 a_{j,1} & \dots & a_{j,i-1} & d
 \end{array}$$

Und im Text ... $\vdots$...

[Buch1/MathForm/dots.tex]

6.3.10 Allgemeine Symbole

In mathematischen Formeln wird eine Vielzahl von Symbolen verwendet. $\text{\LaTeX}$ stellt eine Reihe häufig verwendeter Symbole zur Verfügung, die in Tabelle 6.7 aufgelistet sind. Einige wenige zusätzliche Symbole für den mathematischen Modus definiert das in 7.10 beschriebene Paket `latexsym`. Auch diese Symbole sind in der Tabelle aufgeführt.

Weitere Symbole stellt die $\mathcal{A}\mathcal{M}\mathcal{S}$ im Paket `amssymb` zur Verfügung, das auch unabhängig vom Paket `amsmath` geladen werden kann. Tabelle 13.7 auf Seite 490 zeigt die Symbole, die in diesem Paket zu finden sind.

Tabelle 6.8: Verschiedene Symbole (vgl. auch amsfonts Tab. 13.7, S. 490)

$\Re$ \Re	$\Im$ \Im	ℓ \ell	$\wp$ \wp
$\imath$ \imath	$\jmath$ \jmath	$\hbar$ \hbar	$\aleph$ \aleph
$\top$ \top	$\bot$ \bot	$\neg$ \neg	∂ \partial
$\forall$ \forall	$\exists$ \exists	∇ \nabla	$\triangle$ \triangle
$\emptyset$ \emptyset	∞ \infty	$\parallel$ \parallel	$\backslash$ \backslash
$\flat$ \flat	$\natural$ \natural	$\sharp$ \sharp	$'$ \prime
$\diamondsuit$ \diamondsuit	$\heartsuit$ \heartsuit	$\spadesuit$ \spadesuit	$\clubsuit$ \clubsuit
$\surd$ \surd	$\angle$ \angle		

Zusätzliche Symbole im Grundpaket latexsym

$\mho$ \mho	$\Box$ \Box	$\Diamond$ \Diamond
-------------	-------------	---------------------

6.3.11 Akzente im mathematischen Modus

In der Mathematik werden Akzente verwendet, um die Bedeutung von Symbolen und Variablen zu verändern. Beispielsweise steht $\dot{x}$ in der Regel für die erste und $\ddot{x}$ für die zweite Ableitung einer Funktion $x(t)$, und $\vec{a}$ ist üblicherweise eine Bezeichnung für einen Vektor. Tabelle 6.9 gibt eine Übersicht aller Akzente im mathematischen Modus.

Neben den Akzenten, die in der Regel nur auf einzelne Variablenbezeichnungen und Symbole angewendet werden, kennt $\LaTeX$ auch die folgenden vier – den Akzenten verwandte – Befehle für (kurze) Teilformeln:

```
\widehat{<Teilformel>}
\widetilde{<Teilformel>}
\overline{<Teilformel>}
\underline{<Teilformel>}
```

Tabelle 6.9: Akzente im mathematischen Modus

$\hat{a}$ \hat{a}	$\check{a}$ \check{a}	$\breve{a}$ \breve{a}	$\tilde{a}$ \tilde{a}
$\acute{a}$ \acute{a}	$\grave{a}$ \grave{a}	$\bar{a}$ \bar{a}	$\vec{a}$ \vec{a}
$\dot{a}$ \dot{a}	$\ddot{a}$ \ddot{a}		

`\widehat` ist eine Art `\hat`-Akzent, der sich über mehrere Zeichen erstrecken kann. `\widetilde` entspricht dem `\tilde`-Akzent und kann sich ebenfalls über mehrere Zeichen ausdehnen. Beide sind allerdings nur in bestimmten Grenzen anpassbar. Sie können über beliebige Teilformeln gesetzt werden, über die sie in der Breite – soweit möglich – angepasst werden. `\overline` setzt einen Strich über eine Teilformel und

verallgemeinert so den `\bar`-Akzent. Lediglich `\underline` hat keine direkte Entsprechung bei den Akzenten. Wegen seiner Analogie zu `\overline` wird es trotzdem hier behandelt. Da `\overline` und `\underline` waagrechte Linien setzen, können sie stets in der Länge der Teilformel gesetzt werden.

Ein kurzes Beispiel demonstriert die Akzente und die verwandten Befehle. Die erste Formel in den Zeilen 8–10 zeigt einige Akzente im mathematischen Modus. Die beiden restlichen Formeln sind Beispiele für die vier den Akzzenten verwandten Befehle.

```

8  \[ \dot{x} \quad \ddot{x}
9      \quad \vec{a} \quad \quad
10     \check{m} \quad \]
11  \[
12  \underline{U}_2 = V
13      \quad \quad
14  \underline{U_2} = V
15      \quad \quad
16  U_{\underline{2}} = V
17  \]
18  \[
19  \overline{x^{m+n}}
20      \not=
21  \overline{x^{m+n}}
22      \not=
23  x^{\overline{m+n}}
24  \]
25  \(\ \widehat{AB} \) \hfil
26  \(\ \widehat{ABC} \) \hfil
27  \(\ \widetilde{AB} \) \hfil
28  \(\ \widetilde{ABCD} \) \)

```

$$\begin{array}{cccc}
 \dot{x} & \ddot{x} & \vec{a} & \check{m} \\
 \underline{U}_2 = V & \underline{U_2} = V & U_{\underline{2}} = V & \\
 \overline{x^{m+n}} \neq \overline{x^{m+n}} \neq \overline{x^{m+n}} & & & \\
 \widehat{AB} & \widehat{ABC} & \widetilde{AB} & \widetilde{ABCD}
 \end{array}$$

[Buch1/MathForm/akzent.tex]

Basispakete

In die eigentliche $\text{\LaTeX} 2_{\epsilon}$ -Distribution sind einige grundlegende Pakete integriert – im Unterschied zu den für eine $\text{\LaTeX} 2_{\epsilon}$ -Installation verlangten (‘required’) Paketen, die in den Kapiteln 9 (Hilfreiche Ergänzungen) und 12 ($\mathcal{A}\mathcal{M}\mathcal{S}$ -Pakete) behandelt werden.

Die hier behandelten Basispakete lassen sich in drei Gruppen einteilen:

- $\text{\LaTeX} 2_{\epsilon}$ -Pakete:
Pakete: `alltt`, `doc`, `exscale`, `fix-cm`, `fixltx2e`, `flafter`, `fontenc`, `graphpap`, `ifthen`, `inputenc`, `latexsym`, `makeidx`, `newlfont`, `oldlfont`, `shortvrb`, `showidx`, `syntonly`, `textcomp`, `tracefmt`
- Dokumentstile und Option für den $\text{\LaTeX} 2.09$ Kompatibilitätsmodus, die die gleichen Erweiterungen des Dateinamens tragen wie Pakete in $\text{\LaTeX} 2_{\epsilon}$:

Dokumentstile (Klassen):

`article`, `book`, `letter`, `report`, `slides`

Stiloptionen (Optionen):

`fleqn`, `leqno`, `proc`, `tlenc`

Mit Ausnahme der Stiloption `proc` werden lediglich die gleichnamigen Klassen oder Optionen von $\text{\LaTeX} 2_{\epsilon}$ geladen.

Ein Dokumentstil `proc` oder eine Stiloption `proc` waren im $\text{\LaTeX} 2.09$ -Standard ursprünglich nicht vorgesehen. Erst später wurde eine Stiloption `proc` für den Dokumentstil `article` entwickelt, aus der in $\text{\LaTeX} 2_{\epsilon}$ eine Klasse wurde. Die Stiloption `proc` kann im Kompatibilitätsmodus gegebenenfalls auch als Dokumentstil verwendet werden.

- Pakete, die in $\text{\LaTeX}2_{\epsilon}$ integriert wurden und keine Funktion mehr haben:

Pakete: bezier, openbib

Sie stehen nur zur Verfügung, um alte Dokumente, die vor der Integration diese Pakete verwendet haben, unverändert formatieren zu können.

Im Folgenden wird nur die erste Gruppe behandelt. Eine Beschreibung der Dokumentstile und Stiloptionen der zweiten Gruppe ist nicht erforderlich, da sich ihre Funktion aus den Klassen und Optionen von $\text{\LaTeX}2_{\epsilon}$ ergibt, die an anderer Stelle beschrieben sind (3.1). Bei den beiden Paketen der dritten Gruppe gibt es nichts zu beschreiben.

7.1 Paket alltt

Häufig werden `verbatim`-Umgebungen genutzt, um in Schreibmaschienschrift Listings oder Texte mit Sonderzeichen zu setzen, die andernfalls nur mit Befehlen gesetzt werden können. Insbesondere sind sie auch geeignet, um $\text{\LaTeX}$ -Eingaben wiederzugeben, da in ihnen Befehle und geschweifte Klammern nicht interpretiert werden, aber wenn man nicht gerade eine $\text{\LaTeX}$ -Beschreibung schreibt, sind diese Eigenschaften meist ohne Bedeutung.

Auf der anderen Seite wäre es schön, wenn in einer `verbatim`-ähnlichen Umgebung Befehle und geschweifte Klammern (zur Abgrenzung ihrer Wirkung oder für Argumente) verwendet werden könnten, die Sonderzeichen und der Zeilenumbruch der Eingabe aber beibehalten würden.

Eine Umgebung mit den gewünschten Eigenschaften stellt das Paket `alltt` von Johannes Braams zur Verfügung. Es wird geladen mit

```
\usepackage{alltt}
```

und kennt keine Optionen, die beim Laden angegeben werden könnten.

Die einzige Wirkung des Pakets besteht in der Definition der gewünschten Umgebung `alltt`:

```

\begin{alltt}
  <Textzeile>
  <Textzeile>
  ...
\end{alltt}

```

Die Zeilen in der Umgebung werden in Schreibmaschinenschrift gesetzt. Die Zeilenstruktur der Eingabe wird wie bei `verbatim` übernommen. Alle Sonderzeichen mit Ausnahme von `\`, `{` und `}` werden als normale Zeichen behandelt und als Zeichen gesetzt. Die Zeichen `\`, `{` und `}` haben hingegen ihre übliche Bedeutung und ermöglichen so die Verwendung von Befehlen in dieser Umgebung, beispielsweise um die Schriftart zu wechseln, eine Datei einzulesen und zu setzen oder eine mathematische Formel – auch eine (einzeilige) abgesetzte – einzufügen. Da die Umgebung am Zeilenende einen Zeilenumbruch in der Ausgabe veranlasst, müssen Formeln allerdings in einer einzigen Zeile eingegeben werden. Dabei ist allerdings zu beachten, dass `&` ein normales Zeichen ist und nicht in tabellenartigen Umgebungen zum Übergang von einer Spalte in die nächste genutzt werden kann. Daher können Umgebungen wie `tabular`, `array` oder `eqnarray` nicht wie gewohnt verwendet werden. Außerdem werden `^` und `_` als normale Zeichen behandelt. Zum Hoch- und Tiefstellen müssen daher die Befehle `\sp` und `\sb` verwendet werden.

Das folgende Beispiel zeigt die Arbeitsweise der `alltt`-Umgebung.

```

6  \usepackage{alltt}
7  \begin{document}
8  \begin{alltt}
9  Dies ist {\textrm{\bfseries Text}} in einer
10 \textsf{alltt}-Umgebung. Selbst Formeln wie
11 \(\sqrt{a\sp{2}+b\sp{2}}\) k"onnen verwendet werden.
12
13 Auch abgesetzte Formeln wie
14 \[ \sum\sb{i=1}\sp{\infty} a\sb{i} \ ]
15 sind m"oglich.
16
17 Und noch ein paar Sonderzeichen %$_^#\~<>@"*|&.
18 \end{alltt}
19 \end{document}

```

Dies ist Text in einer
alltt-Umgebung. Selbst Formeln wie
 $\sqrt{a^2 + b^2}$ können verwendet werden.

Auch abgesetzte Formeln wie

$$\sum_{i=1}^{\infty} a_i$$

sind möglich.

Und noch ein paar Sonderzeichen `$_^#~<>@"*|&.`

[Buch1/Basispakete/alltt.tex]

7.2 Paket doc

Dieses Paket spielt eine sehr wichtige Rolle bei den $\text{\LaTeX} 2_{\epsilon}$ -Paketen. Pakete werden in der Regel als Datei mit der Namensendung `.dtx` verteilt. Einerseits können aus dieser Datei mit Hilfe von $\text{\LaTeX}$ die Klassen, Pakete und anderen Dateien generiert werden, die Teil der Verteilung sind, andererseits kann die Datei mit $\text{\LaTeX}$ formatiert werden und erstellt dann die zugehörige Dokumentation.

Das Paket `doc` spielt, wie der Name bereits andeutet, eine zentrale Rolle beim Erstellen der $\text{\LaTeX}$ -Dokumentation. Es wird daher auch von der Dokumentklasse `ltxdoc` verwendet. Hauptaufgabe der Klasse ist dabei das Festlegen des allgemeinen Layouts der Dokumentation. `doc` stellt hingegen alle speziellen Befehle und Umgebungen zur Verfügung, die für das Setzen der Dokumentation aus den oben erwähnten `.dtx`-Dateien von Bedeutung sind. Diese Aufteilung erlaubt es, die Dokumentation auch mit anderen Klassen zu setzen.

Das Paket ist daher für jeden von großer Bedeutung, der Pakete entwickelt und verteilt. Dieser Personenkreis sollte sich die zugehörige Dokumentation genau ansehen und sich bei der Dokumentation seines Pakets darauf stützen.

Für den Anwender sind hingegen die speziellen Befehle und Umgebungen bedeutungslos, auch wenn das Paket selbst unverzichtbar ist, um Dokumentationen von Paketen zu formatieren. Da dieses Buch sich vor allem an Anwender richtet, kann hier auf eine Beschreibung der Befehle und Umgebungen des Pakets verzichtet werden.

7.3 Paket exscale

Die großen mathematischen Zeichen sind in unterschiedlichen Größen in einem einzelnen Font ‚cmex10‘ zu finden, der üblicherweise nur in der Grundgröße 10 pt zur Verfügung steht. Es wurden immer wieder Wünsche laut, den Font auch in anderen Größen verwenden zu können, um eine bessere Anpassung der großen Symbole an die Höhe von Teilformeln zu erreichen.

Dieses Paket von Frank Mittelbach und Dr. Rainer Schöpf erfüllt diesen Wunsch. Neben den Laden des Pakets mit

```
\usepackage{exscale}
```

(Optionen sind nicht vorgesehen) muss allerdings auch der Font in den benötigten Größen verfügbar sein. Es handelt sich dabei um die Größen 7 pt, 8 pt, 9 pt, 10.95 pt, 12 pt, 14.4 pt, 17.28 pt, 20.74 pt und 24.88 pt. Die fehlenden Größen werden von der $\mathcal{A}\mathcal{M}\mathcal{S}$ zwar zur Verfügung gestellt; aber es sind skalierte Fonts und nicht speziell an die Größe angepasste. Aus diesem Grund sind die Autoren von exscale auch der Ansicht, dass sich mit dem Paket die Qualität nicht so verbessert, wie es wünschenswert wäre. Dazu müssten speziell an die Größe angepasste Versionen des Fonts entwickelt werden.

7.4 Pakete fixltx2e und fix-cm

In $\text{\LaTeX}$ gibt es einige wenige Probleme, bei denen bekannt ist, wie sie sich beheben lassen, deren Korrektur aber zu Inkompatibilitäten mit früheren Versionen führen würde. Aus diesem Grund haben die Autoren von $\text{\LaTeX} 2_\epsilon$ sich gegen eine Korrektur von $\text{\LaTeX} 2_\epsilon$ selbst entschieden und bieten die Korrekturen in Form eines Pakets an. Die Autoren sind Frank Mittelbach, David Carlisle und Chris Rowley. Da es nicht sehr wahrscheinlich ist, dass Anwender mit den Problemen konfrontiert werden, wird dieses Paket selten verwendet. Gelegentlich kann man allerdings auch enthaltene Erweiterungen nutzen.

Bei den Fonts liegen einige wünschenswerte Verbesserungen vor, deren automatische Berücksichtigung aber dazu führen würde, dass sich beim Neuformatieren älterer Dokumente der Zeilen- und Seitenumbruch ändern würde. Mit derselben Argumentation wie oben hat Walter Schmidt das Paket fix-cm zur Verfügung gestellt.

7.4.1 fixltx2e

Die Korrekturen zu L^AT_EX 2_ε können durch Laden des Pakets `fixltx2e` aktiviert werden. Es wird in der üblichen Weise mit

```
\usepackage{fixltx2e}
```

geladen.

Die folgende Liste führt die durch das Paket korrigierten Probleme und Änderungen auf Wunsch von Anwendern auf. Weitere Einzelheiten können in der Paketdokumentation nachgelesen werden.

- Bei zweispaltigem Satz mit der Option `twocolumn` der Standardklassen oder mit dem `\twocolumn`-Befehl kann es vorkommen, dass im Fall einer zweispaltigen Abbildung oder Tabelle und einer späteren einspaltigen Abbildung oder Tabelle die einspaltige *vor* der zweispaltigen gesetzt wird. Auch der umgekehrte Fall ist möglich. Abbildungen oder Tabellen würden in diesen Fällen nicht mehr in Reihenfolge.
- Wird bei zweispaltigem Satz in der Kopfzeile `\firstmark` verwendet, so wird die `\firstmark` der zweiten statt die der ersten Spalte verwendet.
- Wird ein Text, der `\@_` enthält, in eine `.aux`-, `.lof`-, `.lot`- oder `.toc`-Datei geschrieben und wird die Datei beim folgenden Formatieren eingelesen und der Text gesetzt, ist an der Stelle `\@_` der Abstand verschwunden.
- `\setlength` führt beim Setzen von Registern wie `\dimen<n>` zu einem Fehler.
- Ein Seitenumbruch direkt bei einem Befehl `\addpenalty` nach einem vorhergehenden `\vspace` führt dazu, dass der untere Seitenrand nicht bündig ist.
- `\fnsymbol` verwendet auch im Textmodus Zeichen aus den mathematischen Fonts, selbst wenn besser passende Zeichen in Textfonts zur Verfügung stehen.
- Ein Gleitobjekt in einem Absatz führt dazu, dass das erste Wort nach dem Gleitobjekt nicht getrennt wird.
- `\em` und `\emph` wechseln von einer geneigten in die aufrechte Schrift, obwohl beispielsweise Kapitälchen geeigneter wären.

Mit `fixltx2e` ist das möglich, indem der Schriftbefehl `\eminnershape` geeignet gesetzt wird, z. B.

```
\renewcommand{\eminnershape}{\scshape}
```

7.4.2 fix-cm

Beim Übergang von den OT1-kodierten CM-Fonts mit maximal 128 Zeichen pro Font zu den T1-kodierten DC- und späteren EC-Fonts und den zugehörigen TS1-kodierten Symbolfonts mit bis zu 256 Zeichen pro Font wurden von Jörg Knappen neue Zeichen in die Fonts aufgenommen und vorhandene überarbeitet. Außerdem wurden fehlende Design-Größen von Fonts ergänzt, die bei den CM-Fonts durch skalieren von Fonts in anderen Design-Größen emuliert werden. Die Überarbeitung von Zeichen und die Verwendung von Fonts in ihrer Design-Größe statt skalierten Fonts hat allerdings den Eindruck verändert, den die Fonts machen. Auch Zeilen- und Seitenumbrüche können bei Verwendung der einen oder anderen Schriftvariante an unterschiedlichen Stellen auftreten.

Das Paket `fix-cm` versucht, das Erscheinungsbild der beiden Schriftvarianten soweit wie möglich anzugleichen. Zusätzlich werden die ‚Computer Modern‘ Fonts in beliebigen Größen verfügbar gemacht, was bei `.pk`-Fonts allerdings zu einer großen Anzahl neu zu generierender Fonts führen kann. Schließlich werden Fonts nur in Design-Größen verwendet, die im Type1-Format zur Verfügung stehen.

Das Laden des Pakets in der üblichen Form ist nicht möglich, da geladene Fonts sich nachträglich nicht mehr ändern lassen – und bereits die `\documentclass`-Anweisung führt dazu, dass Standardfonts geladen werden. Der `\usepackage`-Befehl kann aber erst nach `\documentclass` verwendet werden, und dann die bereits geladenen Standardfonts nicht mehr angepasst werden.

Einen Ausweg bietet der Befehl `\RequirePackage`, der eine ähnliche Funktion wie `\usepackage` hat (er lädt ein Paket allerdings nur, wenn es nicht bereits geladen ist). Dieser Befehl kann **vor** `\documentclass` verwendet werden:

```
\RequirePackage{fix-cm}
\documentclass{...}
```

Das Paket unterstützt die folgenden Schriftfamilien:

- Textfonts `cmr`, `cmss`, `cmtt` und `cmvtt` in OT1-, T1- und TS1-Kodierung;
- von $\text{\LaTeX}$ voreingestellte mathematische Fonts `cmn` in OML- und `cms` in OMS-Kodierung;
- die Fontfamilie `lasy` mit den vom Paket `latexsym` verwendeten Symbolen.

Die Dokumentation weist ausdrücklich darauf hin, dass die folgenden Fontfamilien, die üblicherweise auch zu einer $\text{\TeX}$ -Installation gehören, *nicht* unterstützt werden, obwohl einige von ihnen häufig genutzt werden:

- Spezielle Fontfamilien wie ‚CM Fibonacci‘, ‚CM Dunhill‘ und ähnliche, die beispielsweise zum Experimentieren verteilt werden;
- CM-Textfonts für Zeichensätze wie Kyrillisch, die nicht auf den in Westeuropa gebräuchlichen Buchstaben aufbauen;
- Zusätzliche mathematische Fonts wie die Symbolfonts der $\mathcal{AMS}$, die häufig auch zusammen mit anderen Fontfamilien eingesetzt werden;
- Der mathematische Erweiterungsfont `cmex`, für den gegebenenfalls andere Pakete wie `exscale`, `amsmath` oder `amsmath` zusätzliche Größen bereit stellen.

7.5 Paket flafter

Wird bei Gleitobjekten wie Abbildungen und Tabellen $\text{\texttt{t}}$ bei den Angaben zur Positionierung verwendet, kann das Gleitobjekt an den Anfang der Seite, auf der es aufgebaut wird, verschoben werden. Es wird daher an einer Stelle gesetzt, an der in der Eingabedatei der Aufbau erst noch bevorsteht. Für einzelne Seiten kann man dieses Vorziehen mit einem `\suppressfloats`-Befehl verhindern.

Sollen Gleitobjekte in einem Dokument aber unter keinen Umständen vor der Stelle gesetzt werden, an der sie in der Eingabedatei aufgebaut werden, lässt sich das durch Laden des Pakets `flafter` von Dr. Leslie Lamport, Frank Mittelbach und Chris Rowley mit

```
\usepackage{flafter}
```

verhindern. Die Angabe von Optionen beim Laden ist nicht vorgesehen.

7.6 Paket fontenc

$\LaTeX$ kann unterschiedlich aufgebaute Fonts setzen, die sowohl 128 als auch 256 Zeichen enthalten können. Um einen Font nutzen zu können, muss $\LaTeX$ wissen, welches Zeichen an welcher Position des Fonts steht. Diese Information bezeichnet man als Kodierung des Fonts.

Für von $\LaTeX$ verwendete Kodierungen haben sich Standardbezeichnungen entwickelt. Die Kodierungen haben die folgende Bedeutung:

- T1** Cork-Kodierung für Text, 8-Bit Standard (,Extended $\TeX$ text'-Kodierung);
- T2A, T2B, T2C** Kodierung für kyrillische Fonts (8 Bit);
- TS1** ,Text Companion'-Kodierung;
- OT1** Kodierung nach Donald E. Knuth für Text ($\TeX$ text'-Kodierung);
- OT2** Kodierung der Universität Washington für kyrillische Fonts;
- OT3** Kodierung für die internationalen phonetischen Fonts;
- OT4** Kodierung für die polnische Erweiterung der OT1-Kodierung von Boguslaw Jackowski und M. Ryćko für die polnische Version der Computer Modern und Computer Concrete Fonts (Polnische Erweiterung der , $\TeX$ text'-Kodierung);
- OML** Kodierung nach Donald E. Knuth für Formeltexte ($\TeX$ math italic'-Kodierung);
- OMS** Kodierung nach Donald E. Knuth für mathematische Symbole ($\TeX$ math symbol'-Kodierung);
- OMX** Kodierung nach Donald E. Knuth für Erweiterungssymbole (große Symbole);
- Lxx** Lokale (individuelle) Kodierung mit der Bezeichnung *xx*;
- LCY** Lokale Kodierung für kyrillische Fonts;
- x2** Spezielle Kodierung für kyrillische Fonts;
- U** Unbekannte Kodierung.

Nicht alle angegebenen Kodierungen sind auch Teil einer $\TeX$ -Distribution. Typischerweise verfügbar sind T1, TS1, OT1, OMS, OML und OT4.

Zum Laden der Kodierung(en), die $\LaTeX$ beim Satz verwenden kann, haben Johannes Braams, David Carlisle, Alan Jeffrey, Frank Mittelbach, Chris Rowley und Dr. Rainer Schöpf das Paket fontenc entwickelt. Beim Laden mit

```
\usepackage[<Kodierung|,<Kodierung>]...]{fontenc}
```

können eine oder mehrere zu verwendende Kodierungen aus der Liste oben angegeben werden, sofern sie verfügbar sind. Die zuletzt angegebene Kodierung wird als Standardkodierung für die Fonts des Dokuments verwendet.

Ein typischer Befehl zum Laden des Pakets und zur Auswahl der EC-Fonts für das Dokument ist die Zeile

```
\usepackage[TS1,T1]{fontenc}
```

in der Präambel, die die Cork-Kodierung als Standard wählt und bei Bedarf den Zugriff auf die zugehörigen Ergänzungsfonts gewährt, die eine Vielzahl von Symbolen enthalten.

Befehle definiert das Paket nicht, da die Befehle zum Aktivieren einer Kodierung und zum Aufbau von Kodierungsdateien Teil von $\LaTeX 2_{\epsilon}$ selbst sind. Zum Aktivieren einer Kodierung dient der Fontbefehl

```
\fontencoding{<Kodierung>}
```

von $\LaTeX 2_{\epsilon}$. In der Regel ist ein solcher Befehl Teil eines Makros zum Setzen von Sonderzeichen. Anwender verwenden den Befehl nur in Ausnahmefällen.

7.7 Paket graphpap

Dieses Paket von Dr. Leslie Lamport definiert ein zusätzliches Objekt für die `picture`-Umgebung. Bei diesem Objekt handelt es sich um ein rechteckiges Gitter, wie man es vom Millimeterpapier kennt.

Um das Objekt verwenden zu können, muss das Paket graphpap mit dem folgenden Befehl geladen werden:

```
\usepackage{graphpap}
```

In einer `picture`-Umgebung kann dann mit einem Befehl ein Gitter gesetzt werden, das am linken und am unteren Rand mit den zugehörigen x - bzw. y -Koordinaten beschriftet wird. Der Befehl zum Setzen des Gitters hat die Form

```
\graphpaper[<Num>](<Lux>,<Luy>)(<Breite>,<Höhe>)
```

und wird in der gleichen Weise wie ein `\circle`-Befehl für einen Kreis verwendet, um das Gitter im Bild zu positionieren. Die Gitterkoordinaten entsprechen den Koordinaten der `picture`-Umgebung, nur dass der Ursprung in der Regel verschoben ist. Die Angaben erfolgen also in Einheiten von `\unitlength`.

Referenzpunkt für die Positionierung ist der Ursprung $(0,0)$ des Gitters, selbst wenn er nicht im Bereich des Gitters liegt. Ein `\put`-Befehl legt also die Position des *Gitterursprungs* fest. Entsprechendes gilt im Fall eines `\multiput`-Befehls.

Alle Argumente des Befehls – `<Num>`, `<Lux>`, `<Luy>`, `<Breite>` und `<Höhe>` – müssen *ganzzahlig* sein; die Angabe nicht ganzzahliger Werte – ansonsten in `picture`-Umgebungen durchaus gebräuchlich – führt beim Befehl `\graphpaper` zu einem fehlerhaften Gitter.

Den Koordinaten $(\text{<Lux>}, \text{<Luy>})$ wird die linke untere Ecke des Gitters zugeordnet. `<Breite>` ist die Breite und `<Höhe>` die Höhe des Gitters jeweils in den Einheiten der `picture`-Umgebung. Das optionale Argument `<Num>` ist der Abstand der Gitterlinien in `picture`-Einheiten. Voreingestellt ist für `<Num>` der Wert 10. Jede fünfte Gitterlinie wird dicker gezeichnet und mit dem zu ihr gehörenden Gitterwert beschriftet.

Der Referenzpunkt $(0,0)$ eines mit `\put (<Px>,<Py>)` positionierten Gitters liegt im Fall positiver Werte `<Lux>` Einheiten links und `<Luy>` unterhalb dieser Ecke. Die linke untere Ecke des Gitters hat die `picture`-Koordinaten $(\text{<Px>}+\text{<Lux>}, \text{<Py>}+\text{<Luy>})$. Die rechte obere Ecke des Gitters mit den Gitterkoordinaten $(\text{<Lux>}+\text{<Breite>}, \text{<Luy>}+\text{<Höhe>})$ entspricht den `picture`-Koordinaten

$$(\text{<Px>}+\text{<Lux>}+\text{<Breite>}, \text{<Py>}+\text{<Luy>}+\text{<Höhe>})$$

Generell entsprechen den Gitterkoordinaten (x,y) eines mit `\put (<Px>,<Py>)` positionierten Gitters die `picture`-Koordinaten $(\text{<Px>}+\text{x}, \text{<Py>}+\text{y})$.

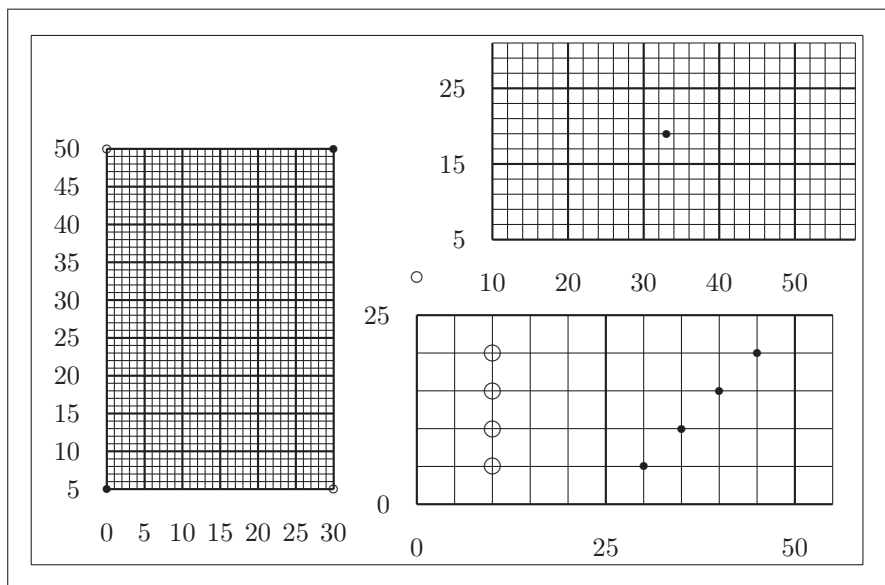
Das folgende Beispiel zeigt eine `graphpap`-Umgebung mit dieser Erweiterung.

```
6 \usepackage{graphpap}
7 \begin{document}
8 \setlength{\unitlength}{1mm}
```

```

9  \begin{picture}(110,70)
10 % Rahmen:
11 \multiput( 0, 0)( 0,70){2}{\line(1,0){110}}
12 \multiput( 0, 0)(110, 0){2}{\line(0,1){ 70}}
13 %
14 \put( 10, 5){\graphpaper[1](0,5)(30,45)}
15 \multiput( 10,10)( 30,45){2}{\circle*{1}}
16 \multiput( 40,10)(-30,45){2}{\circle{1}}
17 %
18 \put( 51,38){\graphpaper[2](10,5)(48,26)}
19 \put( 51,38){\circle{1.5}}
20 \put( 84,57){\circle*{1}}
21 %
22 \put( 51, 8){\graphpaper[5](0,0)(55,25)}
23 \multiput( 61,13)( 0, 5){4}{\circle{2}}
24 \multiput( 81,13)( 5, 5){4}{\circle*{1}}
25 \end{picture}
26 \end{document}

```



[Buch1/Basispakete/graphpap.tex]

7.8 Paket ifthen

Gelegentlich möchte man auch als Anwender Teile eines Dokuments abhängig von bestimmten Einstellungen unterschiedlich setzen. Solange es

nur sich um die Anpassung des Layouts handelt, können die Änderungen mit Hilfe der Layout-Parameter von $\LaTeX$ vorgenommen werden.

Was aber, wenn man auf Abbildungen oder Tabellen mit *unterschiedlichen Texten* verweisen will, je nachdem, ob die Abbildung oder Tabelle in einem zweiseitigen Layout sichtbar ist, oder ob der Leser nur wenige oder sogar viele Seiten vor- oder zurückblättern muss? Dieses spezielle Problem löst übrigens das Paket `varioref` (s. 9.21, S. 301). Vielleicht möchte man aber auch in einem zweiseitigen Layout bei schmalen Abbildungen die ‚Bildunterschrift‘ außen neben das Bild setzen, also auf geraden Seiten links und auf ungeraden rechts vom Bild.

Will man derartige Abläufe automatisieren, muss man Bedingungen überprüfen und je nach Ergebnis unterschiedlich reagieren können. Die dazu notwendigen Befehle stellt das Paket `ifthen` von David Carlisle zur Verfügung. Es wird mit dem Befehl

```
\usepackage{ifthen}
```

geladen und kann im Anschluss daran bereits in der Präambel genutzt werden, sofern keine Ausgaben erzeugt werden.

Das Paket stellt zwei Grundbefehle zur Verfügung. Der wichtigere der beiden berücksichtigt – abhängig von einer Bedingung – eine von zwei Alternativen:

```
\ifthenelse{<Bedingung>}{<Wahr-Code>}{<Falsch-Code>}
```

<Bedingung> ist ein logischer Ausdruck, und wenn die Bedingung zutrifft (wahr ist), wird der als *<Wahr-Code>* angegebene Teil verwendet. Andernfalls wird der als *<Falsch-Code>* bezeichnete Teil berücksichtigt. *<Bedingung>* darf nicht fehlen, aber die Alternativen können leer sein, wenn in ihrem Fall nichts geschehen soll.

```
\ifthenelse{1=1}{}{}
```

ist für $\LaTeX$ daher eine korrekter Leerbefehl. Wie *<Bedingung>* aufgebaut sein muss, wird im folgenden genauer betrachtet.

Das folgende Beispiel zeigt die Verwendung des Befehls `\ifthenelse`. In den Zeilen 7–15 werden in Anlehnung an ein Beispiel aus Goossens et al. (1995) der Befehl `\ord` definiert, der englische Ordnungszahlen setzt. Dabei werden drei ineinander geschachtelte `\ifthenelse`-Befehle verwendet, wobei die inneren `\ifthenelse` jeweils im Zweig *<Falsch-Code>* verwendet werden. Als Argument von `\ord` ist jeweils ein Zähler anzugeben, beispielsweise `section` oder `page`. Um die Wirkung zu zeigen, werden nach `\begin{document}` unter Verwendung des `section`-Zählers die ersten vier Ordnungszahlen gesetzt.

```

6 \usepackage{ifthen}
7 \newcommand{\ord}[1]{%
8   \the\value{#1}%
9   \ifthenelse{\value{#1} = 1}%
10    {$^{\mbox{\scriptsize st}}$}%
11    {\ifthenelse{\value{#1} = 2}%
12     {$^{\mbox{\scriptsize nd}}$}%
13     {\ifthenelse{\value{#1} = 3}%
14      {$^{\mbox{\scriptsize rd}}$}%
15      {$^{\mbox{\scriptsize th}}$}}}%
16 \begin{document}
17 \setcounter{section}{1}\ord{section} chapter
18
19 \setcounter{section}{2}\ord{section} chapter
20
21 \setcounter{section}{3}\ord{section} chapter
22
23 \setcounter{section}{4}\ord{section} chapter
24 \end{document}

```

1st chapter

2nd chapter

3rd chapter

4th chapter

[Buch1/Basispakete/ifthen1.tex]

Der zweite Grundbefehl dient dazu, einen als Argument angegebenen $\LaTeX$ -Code solange zu wiederholen, wie die *<Bedingung>* erfüllt ist:

```
\whiledo{<Bedingung>}{<Code>}
```

<Bedingung> hat dabei die gleiche Form wie beim `\ifthenelse`-Befehl. Um die Schleife zu beenden, muss der $\LaTeX$ -Code *<Code>* die Bedingung so verändern, dass sie nicht mehr zutrifft. Das kann beispielsweise durch Ändern eines Zählers geschehen.

Im folgenden Beispiel wird `\whiledo` verwendet, um in den Zeilen 8–12 den Befehl `\countdown` zu definieren, der von einem als Argument angegebenen Wert rückwärts bis zur Null zählt. Die Verwendung des Befehls zeigen die Zeilen 14 und 16.

```

6 \usepackage{ifthen}
7 \newcounter{cnt}
8 \newcommand{\countdown}[1]{%
9   #1\setcounter{cnt}{#1}%
10  \whiledo{\thecnt>0}{%
11    \addtocounter{cnt}{-1}%
12    \dots\thecnt}}
13 \begin{document}
14 \countdown{3}
15
16 \countdown{5}
17 \end{document}

```

3...2...1...0

5...4...3...2...1...0

[Buch1/Basispakete/whiledo.tex]

Offen ist noch die Frage, wie *<Bedingung>* auszusehen hat. Es wurde bereits gezeigt, dass zwei Zahlenwerte auf Gleichheit getestet werden können. Statt dessen kann man auch prüfen, ob der erste Zahlenwert kleiner ist als der zweite, oder ob er größer ist. Vor allem kann man aber statt mit einer festen Zahl auch mit dem Wert `\value{<Zähler>}` eines Zählers vergleichen.

Zahlenwerte – ganze Zahlen oder mit `\value` Werte von Zählern – können also mit den drei Operatoren

`<` `>` `=`

verglichen werden.

Die gleichen Operatoren lassen sich zum Vergleichen von Längen verwenden. Allerdings muss in diesem Fall $\LaTeX$ mit dem Befehl

```
\lengthtest{<Längenvergleich>}
```

mitgeteilt werden, dass Längen und nicht Zahlenwerte verglichen werden. Bei den Längen kann es sich sowohl um explizite Längen wie `3cm` als auch um Längen wie `\textwidth` handeln.

```
\lengthtest{\textwidth>10cm}
```

ist wahr, wenn die Textbreite größer ist als 10 cm.

Zahlenwerte lassen sich außerdem mit

```
\isodd{<Zahlenwert>}
```

daraufhin überprüfen, ob sie gerade sind. Die Bedingung

```
\isodd\value{page}
```

ist also wahr, wenn die Seitennummer ungerade ist, und falsch für eine gerade Seitennummer. Entsprechend ist

```
\isodd\pageref{<Marke>}
```

wahr, wenn sich `Cmdlabel{<Marke>}` auf eine Seite mit ungerader Nummer befindet.

Auch Zeichenketten können auf Übereinstimmung getestet werden. Das Paket `ifthen` stellt dazu den Befehl

```
\equal{<Zeichenkette1>}{<Zeichenkette2>}
```

zur Verfügung. Wird anstelle einer Zeichenkette ein Befehl angegeben oder enthält eine Zeichenkette einen Befehl, so expandiert $\text{\LaTeX}$ den Befehl vollständig (gegebenenfalls über mehrere Stufen), bevor die beiden Zeichenketten verglichen werden. Dabei ist aber zu beachten, dass Paare geschweifeter Klammern, die nicht ein Argument eines Befehls einschließen, Teil der Zeichenkette sind. Daher hat die Bedingung

```
\equal{ab{c}de}{abcde}
```

den Wahrheitswert `false` (falsch).

Schließlich kann man prüfen, ob ein Befehl definiert ist oder nicht. Dieser Test hat die Form

```
\isundefined{<Befehl>}
```

und liefert den Wahrheitswert `true` (wahr), wenn der als Argument angegebene Befehl *nicht* definiert ist. Demzufolge hat

```
\isundefined{\section}
```

beispielsweise im Fall der Dokumentklasse `article` den Wahrheitswert `false`, hingegen bei der Dokumentklasse `letter` den Wahrheitswert `true`.

Neben diesen Bedingungen kennt das Paket `ifthen` auch logische Variablen, deren aktueller Wert mit

```
\boolean{<Variablenname>}
```

abgefragt werden kann. $\text{\LaTeX}$ verwendet selbst solche logischen Variablen, und diese können mit `\boolean` auch abgefragt werden. Eine vollständige Liste dieser Variablen wäre sehr umfangreich, und daher werden hier nur einige wichtige Variablen kurz beschrieben. Die Auswahl der wichtigen logischen Variablen wurde aus Goossens et al. (1995) übernommen.

<code>hmode</code>	<code>true</code> , während Text horizontal gesetzt wird, beispielsweise in einem Absatz
<code>vmode</code>	<code>true</code> , wenn $\TeX$ Elemente vertikal zusammenfügt, beispielsweise <i>zwischen</i> zwei Absätzen
<code>mmode</code>	<code>true</code> , wenn eine Formel gesetzt wird
<code>@twoside</code>	<code>true</code> , wenn ein Dokument zweiseitig gesetzt wird
<code>@twocolumn</code>	<code>true</code> , wenn das Dokument mit der <code>twocolumn</code> -Option gesetzt wird oder bei zweispaltiger Ausgabe mit dem <code>\twocolumn</code> -Befehl (nicht aber in einer <code>multicols</code> -Umgebung)
<code>@firstcolumn</code>	<code>true</code> , wenn <code>@twocolumn</code> wahr ist und gerade die erste (linke) Spalte gesetzt wird
<code>@newlist</code>	<code>true</code> , wenn eine Listenumgebung begonnen wurde bis zum Beginn des Texts nach dem ersten <code>\item</code> -Befehl (der erste <code>\item</code> -Befehl setzt den Wert auf <code>false</code>)
<code>@inlabel</code>	<code>true</code> , solange das optionale Argument eines <code>\item</code> -Befehls bearbeitet wird
<code>@noskipsec</code>	<code>true</code> nach einer Überschrift, die nicht durch einen vertikalen Abstand vom nachfolgenden Text getrennt ist, bis zum Anfang des nachfolgenden Texts (etwa nach <code>\paragraph</code> , aber falsch nach <code>\section</code>)

Bei den letzten sechs logischen Variablen wird auf die Besonderheit hingewiesen, dass ihre Namen mit einem Klammeraffen beginnen.

Außerdem hat der Anwender die Möglichkeit, eigene logische Variablen zu vereinbaren und ihren Wert zu setzen. Dazu stehen die folgenden Befehle zur Verfügung:

```
\newboolean{<Variablenname>}
```

Vereinbart eine *neue* logische Variable `<Variablenname>`; führt zu einem Fehler, falls die logische Variable bereits vereinbart ist

```
\provideboolean{<Variablenname>}
```

Stellt sicher, dass die logische Variable `<Variablenname>` vereinbart ist

```
\setboolean{<Variablenname>}{<Wert>}
```

Weist der logischen Variablen `<Variablenname>` den Wert `true` oder `false` zu. Zuweisen anderer Werte führt zu einem Fehler.

Die beschriebenen Bedingungen können aber nicht nur isoliert getestet werden, sondern sie können zu komplexen logischen Ausdrücken kombiniert werden, deren Wahrheitswert von `\ifthenelse` überprüft wird. Die folgenden Operatoren können beim Aufbau logischer Ausdrücke verwendet werden:

`\and` true, falls sowohl der vorhergehende als auch der nachfolgende logische Ausdruck den Wahrheitswert `true` hat, andernfalls `false`

`\or` true, falls der vorhergehende oder der nachfolgende logische Ausdruck den Wahrheitswert `true` hat, `false` nur, wenn beide den Wahrheitswert `false` haben

`\not` negiert den Wahrheitswert des nachfolgenden logischen Ausdrucks

Die Reihenfolge, in der die Operatoren ausgewertet werden, kann mit Hilfe von Klammern

```
\( <Logischer Ausdruck> \)
```

festgelegt werden, die von innen nach außen ausgewertet werden.

7.9 Paket inputenc

T_EX ist ein Textsatzsystem, das auf den unterschiedlichsten Rechnern eingesetzt wird. Daher muss T_EX mit Eingabedateien arbeiten, die auf einem beliebigen dieser Rechner erstellt worden sind. Das Problem dabei ist, dass Rechner unterschiedliche Kodierungen („encodings“) für die über die Tastatur eingegebenen Zeichen verwenden. In der Regel lässt sich die verwendete Kodierung am Rechner einstellen. Nicht alle Windows-Rechner – und entsprechendes gilt für UNIX-Rechner – verwenden daher die gleiche Kodierung.

Auf der anderen Seite sieht ein Programm wie T_EX nur den Code, in den der Rechner ein eingegebenes Zeichen umgesetzt hat, und nicht, welche Taste gedrückt wurde. Das Problem ist allerdings bei weitem nicht so groß, wie es zunächst den Anschein hat, da alle anderen Kodierungen im Bereich 32–127 mit der minimalen ASCII-Kodierung übereinstimmen.

Dieser minimale ASCII-Zeichensatz enthält die im Englischen verwendeten Buchstaben, die Ziffern sowie wichtige Akzente und Sonderzeichen.

Er ist völlig ausreichend, um jedes $\text{T}_{\text{E}}\text{X}$ - und $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Dokument zu setzen. Andere Eingabekodierungen erlauben lediglich die vereinfachte Eingabe zusätzlicher Zeichen über die Tastatur und verbessern damit die Lesbarkeit der Eingabedateien. Der Preis dafür ist allerdings, dass die zusätzlichen Zeichen nicht einheitlich kodiert sind und $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ daher die verwendete Kodierung mitgeteilt werden muss.

Auch die Übertrag auf einen Rechner mit anderer Kodierung ist etwas komplizierter: Entweder müssen die Dateien *binär* übertragen werden und können dann unverändert mit $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ formatiert werden; Nicht-ASCII-Zeichen werden am Zielrechner aber möglicherweise nicht korrekt angezeigt. Oder man überträgt sie als Textdateien, bei denen die Kodierung der Zeichen während der Übertragung an den Zielrechner angepasst wird; dann werden alle Zeichen am Zielrechner korrekt dargestellt, aber $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ muss angewiesen werden, eine andere Kodierung – die des Zielrechners – zu berücksichtigen.

$\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ verwendet zum Festlegen der in der Eingabedatei verwendeten Kodierung das Paket `inputenc` von Alan Jeffrey und Frank Mittelbach. Es wird wie üblich mit

```
\usepackage[<Kodierung>]{inputenc}
```

geladen. Die Kodierung der Eingabedatei(en) wird als Option angegeben.

Mehrsprachige Dokumente kann man sogar mit einem jeweils an die Sprache angepassten Zeichensatz (also angepasster Kodierung) schreiben. Der Befehl zum Wechsel der verwendeten Eingabekodierung lautet

```
\inputencoding{Kodierung}
```

und kann überall innerhalb eines Dokuments verwendet werden.

Die folgenden Kodierungen sind üblicherweise Teil der $\text{T}_{\text{E}}\text{X}$ -Distribution und können als Option beim Laden von `inputenc` oder beim Befehl `\inputencoding` als Argument angegeben werden:

ascii ASCII-Kodierung mit sichtbaren Zeichen nur im Bereich 32–127 (0–31 sind Steuerzeichen);
Minimaler Zeichensatz mit den im Englischen gebräuchlichen Zeichen und Akzenten, aus denen sich nationale Sonderzeichen wie im Deutschen Umlaute und Eszett zusammensetzen lassen.

ansinew	Unter Windows 3.1 häufig verwendete ANSI-Kodierung, eine Erweiterung von ISO Latin 1;
applemac	Kodierung des Zeichensatzes am MacIntosh;
cp437	IBM-Codepage 437 (DOS Latin US);
cp437de	IBM-Codepage 437 mit deutschen Sonderzeichen (DOS Latin DE);
cp850	IBM-Codepage 850 für Westeuropa (DOS Latin 1);
cp852	IBM-Codepage 852 für Osteuropa (DOS Latin 2);
cp858	IBM Codepage 850 ergänzt um Euro-Symbol;
cp865	IBM-Codepage 865 für Norwegen;
cp1250	MS Windows-Codepage 1250 für Zentral- und Osteuropa;
cp1252	MS Windows-Codepage 1252 für Westeuropa;
cp1257	MS Windows-Codepage 1257 für die baltischen Staaten;
decmulti	Kodierung des multinationalen DEC-Zeichensatzes;
latin1	ISO 8859-1 (Latin-1) Kodierung (Westeuropa);
latin2	ISO 8859-2 (Latin-2) Kodierung (Zentral- und Osteuropa);
latin3	ISO 8859-3 (Latin-3) Kodierung (Westeuropa und Türkei);
latin4	ISO 8859-4 (Latin-4) Kodierung (Nordeuropa);
latin5	ISO 8859-9 (Latin-5) Kodierung (Türkei);
latin9	ISO 8859-15 (Latin-9) Kodierung (im wesentlichen Latin 1 mit Euro-Symbol);
latin10	ISO 8859-16 (Latin-10) Kodierung (Westeuropa und Rumänien);
macce	Kodierung des zentraleuropäischen Zeichensatzes am MacIntosh;

- next** Kodierung des Zeichensatzes an Next-Rechnern;
- utf8** eine Unicode-Kodierung (unter Linux Standard).

Heutzutage dürften `latin1` und `utf8` die am weitesten verbreiten Kodierungen unter Linux und Windows im deutschsprachigen oder sogar westeuropäischen Raum sein.

Das Paket definiert weitere Befehl, die zum Aufbau von Kodierungsdateien dienen und auch zum Festlegen der vorgestellten Kodierungen verwendet wurden. Für den Anwender sind sie hingegen nicht von Interesse. Auf ihre Beschreibung wird daher verzichtet.

7.10 Paket latexsym

Dieses Paket von Frank Mittelbach stellt dem Anwender einige mathematische Symbole zur Verfügung, die standardmäßig nicht angeboten werden, da sie relativ selten benutzt werden. Sie fehlen auch in den von $\text{\LaTeX}$ automatisch geladenen Standardfonts. Um die wenigen Zeichen nutzen zu können, müssen zusätzliche Fonts geladen werden. Verwendet man eines der Paket `amsmath` (s. 13.3.1, S. 484) oder `amssymb` (s. 13.3.2) sind die Zeichen schon verfügbar, da auch die Fonts der $\mathcal{AMS}$ sie enthalten.

Werden aber nur diese wenigen Zeichen zusätzlich benötigt, kann man das Paket mit

```
\usepackage{latexsym}
```

laden und auf die Fonts der $\mathcal{AMS}$ verzichten. Optionen können beim Laden nicht angegeben werden.

Tabelle 7.1 zeigt die zusätzlichen Zeichen. Sie sind je nach ihrer Bedeutung als Operator, Relation oder Symbol auch in den Tabellen 6.2, 6.3 und 6.7 bei der Beschreibung des mathematischen Modus aufgeführt.

Tabelle 7.1: Symbole im Paket latexsym

$\triangleleft$ <code>\lhd</code>	$\triangleright$ <code>\rhd</code>	$\trianglelefteq$ <code>\unlhd</code>	$\trianglerighteq$ <code>\unrhd</code>
$\Join$ <code>\Join</code>	$\leadsto$ <code>\leadsto</code>	$\sqsubset$ <code>\sqsubset</code>	$\sqsupset$ <code>\sqsupset</code>
$\Box$ <code>\Box</code>	$\Diamond$ <code>\Diamond</code>	$\mho$ <code>\mho</code>	

7.11 Pakete makeidx und showidx

Auch ohne Unterstützung von Paketen stellt L^AT_EX die Befehle zur Verfügung, die zum Aufbau eines Stichwortverzeichnisses benötigt werden. In der Präambel ist der Befehl `\makeindex` zu verwendenden, der eine Datei mit der Namensendung `.idx` zum Sammeln der Stichworte und der jeweiligen Seitenzahl bereitstellt und damit die Informationen für das Stichwortverzeichnis neu aufbaut. Nach `\begin{document}` können mit dem `\index`-Befehl Stichworte zusammen mit der aktuellen Seite in die `.idx`-Datei geschrieben werden. Die `.idx`-Datei kann nach dem Formatieren beispielsweise mit MakeIndex zu einem Stichwortverzeichnis in einer Datei mit der Namensendung `.ind` aufbereitet werden, die beim nächsten Formatieren mit `\input` eingelesen werden kann und das Stichwortverzeichnis setzt.

Die beiden hier behandelten Pakete `makeidx` und `showidx` von Johannes Braams, David Carlisle, Alan Jeffrey, Leslie Lamport, Frank Mittelbach, Chris Rowley und Dr. Rainer Schöpf erleichtern das Setzen des Stichwortverzeichnisses (`makeidx`) und verbessern die Kontrolle der Einträge mit dem `\index`-Befehl (`showidx`). Mit `makeidx` sind außerdem Querverweise im Stichwortverzeichnis möglich.

7.11.1 makeidx

Dieses Paket erleichtert dem Anwender das Setzen eines Stichwortverzeichnisses. Zusätzlich definiert es Befehle, die Querverweise im Stichwortverzeichnis erlauben. Das Paket wird mit

```
\usepackage{makeidx}
```

laden. Optionen sind nicht vorgesehen. Nach dem Laden können im Dokument die folgenden Befehle verwendet werden:

```
\printindex
```

Liest eine beispielsweise mit MakeIndex aufbereitete Indexdatei mit der Dateinamenserweiterung `.ind` ein, falls sie vorliegt, und setzt damit das Stichwortverzeichnis

```
\see{<Ziel>}{<Seite>}
```

Verweist auf einen anderen Eintrag `<Ziel>` im Stichwortverzeichnis. Das zweite Argument `<Seite>` wird in der Regel ignoriert; es wird beim Aufbereiten des Stichwortverzeichnisses automatisch angehängt und muss daher vorgesehen sein.

`\seealso{<Ziel>}{<Seite>}`

Verweist auf einen zusätzlichen Eintrag `<Ziel>` im Stichwortverzeichnis. Das zweite Argument `<Seite>` wird in der Regel ignoriert; es wird beim Aufbereiten des Stichwortverzeichnisses automatisch angehängt und muss daher vorgesehen sein.

`\seename`

Text, der von `\see` zum Verweisen auf einen anderen Eintrag vor `<Ziel>` gesetzt wird

Voreinstellung: `see`

`\alsiname`

Text, der von `\seealso` zum Verweisen auf einen anderen Eintrag vor `<Ziel>` gesetzt wird

Voreinstellung: `see also`

Das folgende Beispiel zeigt dies:

```

12 \makeatother
13 \usepackage{makeidx}
14 \renewcommand{\seename}%
15     {siehe}
16 \renewcommand{\alsiname}%
17     {siehe auch}
18 \makeindex
19 \begin{document}
20 \section{Philosophie}
21 Der Weg\index{Weg} ist
22 das Ziel\index{Ziel}.
23
24 Eine Stra"se%
25 \index{Stra"ss{e|see{Weg}}
26 ist ein befestigter Weg.
27 \index{Weg|seealso{Ziel}}
28
29 %% \printindex %% hier:
30 \input{makeidx.ind}
31 \end{document}

```

1 Philosophie

Der Weg ist das Ziel.
Eine Straße ist ein befestigter Weg.

[Buch1/Basispakete/makeidx.tex]

7.11.2 showidx

Der eigentliche Aufwand beim Aufbau eines Stichwortverzeichnisses besteht darin, die Einträge an den richtigen Stellen vorzunehmen. Hinzu kommt, dass unterschiedliche Schreibweisen – und das können be-

reits führende oder angehängte Leerzeichen sein – zu verschiedenen Einträgen im Stichwortverzeichnis führen, die nicht zusammengefasst werden.

Dieses Paket soll dabei helfen, die richtigen Stellen für die `\index`-Befehle zu finden und Probleme zu vermeiden. Das Paket wird ohne Angabe von Optionen mit

```
\usepackage{showidx}
```

geladen und verwendet `\flushbottom` als Voreinstellung. Es bewirkt, dass jeder `\index`-Befehl sein Argument zusätzlich in den Seitenrand setzt. Damit lassen sich die Einträge in das Stichwortverzeichnis an Ort und Stelle überprüfen.

Das folgende Beispiel zeigt, wie das Paket wirkt. Es zeigt auch, dass dabei Probleme auftreten können. In den Zeilen 10–12 wird ein Befehl `\Bef` zum Setzen von Befehlen definiert und gleichzeitig mit `\index` zwei Einträge in das Stichwortverzeichnis vornimmt. `showidx` zeigt bei Verwendung dieses Befehls in Zeile 17 den vorgenommenen Eintrag korrekt an. Werden in den Zeilen 21–23 aber die gleichen Befehle wie bei der Definition von `\Bef` direkt verwendet, ist die Ausgabe des Arguments in den Seitenrand nicht wie es zu erwarten wäre. Aus Gründen der Vereinheitlichung ist allerdings die Verwendung eines Befehls wie `\Bef` ohnehin ratsam.

```

11 \usepackage{makeidx,showidx}
12 \makeindex
13 \newcommand{\Bef}[1]{\texttt{\textbackslash#1}%
14   \index{#1@\texttt{\textbackslash#1}}%
15   \index{Befehle!#1@\texttt{\textbackslash#1}}}
16 \begin{document}
17 \section{Beispiel f"ur \textsf{show\textbackslash-id}}
18
19 Das Paket\index{Paket} setzt das Argument des
20 \Bef{index}-Befehls in Form einer
21 Randnotiz\index{Randnotiz}.
22
23 Wird jedoch ein komplizierterer Eintrag allerdings
24 ohne Befehl wie in \texttt{\textbackslash index}%
25   \index{index@\texttt{\textbackslash index}}%
26   \index{Befehle!index@\texttt{\textbackslash index}}
27 gesetzt, sind Probleme nicht ausgeschlossen.
28 \end{document}
```

1 Beispiel für show-idx

Das Paket setzt das Argument des `\index`-Befehls in Form einer Randnotiz.
Wird jedoch ein komplizierterer Eintrag allerdings ohne Befehl wie in `\index` gesetzt, sind Probleme nicht ausgeschlossen.

1

```
Paket
index@\index
Befehle!index@\index
Randnotiz
index@\texttt\textbackslash
index
Befehle!index@\texttt\textbackslash
index
```

[Buch1/Basispakete/showidx.tex]

7.12 Pakete newfont und oldfont

Im Laufe der Entwicklung von $\text{\LaTeX}$ 2.09 zu $\text{\LaTeX}$ 2_ε gab es Änderungen bei den Befehlen zur Fontauswahl und in ihrem Verhalten. Die beiden hier vorgestellten Pakete von Frank Mittelbach emulieren das Verhalten der ursprünglichen Fontbefehle von $\text{\LaTeX}$ 2.09 (oldfont) und der ersten Version des NFSS (New Font Selection Scheme) während der Zeit des Übergangs von $\text{\LaTeX}$ 2.09 zu $\text{\LaTeX}$ 2_ε (newfont).

7.12.1 oldfont

In $\text{\LaTeX}$ 2.09 setzten die aus zwei Buchstaben bestehenden Fontbefehle sich gegenseitig außer Kraft, da es ursprünglich keine Fontattribute gab. Sie wählten daher jeweils einen speziellen Font (mit festen Fontattributen) aus. Das Laden des Pakets mit

```
\usepackage{oldfont}
```

bewirkt, dass sich Fontbefehle wie `\rm` oder `\sf` sich wie bei $\text{\LaTeX}$ 2.09 verhalten. Option sind für das Paket nicht vorgesehen.

Da in $\text{\LaTeX}$ 2.09 nur diese Fontbefehle zur Verfügung standen, mussten sie auch im mathematischen Modus verwendet werden, wenn in Formeln Text einzufügen war. Das Paket erlaubt die Verwendung der Befehl in Formeln ebenfalls.

Schließlich wird das Paket `latexsym` (s. 7.10) geladen.

Die internen Fontbefehle von $\text{\LaTeX}$ 2.09 wie `\tenrm` oder `\elvsvf` werden jedoch nicht definiert. Dokumente, die die internen Fontbefehle verwenden, können das Paket `rawfonts` (s. 9.15) laden, das die internen Fontbefehle zur Verfügung stellt.

7.12.2 newfont

In der ersten Version des NFSS verhielten sich die aus zwei Buchstaben bestehenden Fontbefehle wie die aktuellen Befehle zum Ändern von Fontattributen. Der Befehl `\tt` hatte die gleiche Wirkung wie der aktuelle `\ttfamily`, und `\sc` verhielt sich wie `\scshape`. Durch Laden des Pakets `newfont` mit

```
\usepackage{newfont}
```

lässt sich dieses Verhalten emulieren. Paketoptionen sind nicht vorgesehen.

Gegebenenfalls kann `newfont` auch genutzt werden, um kürzere Befehle für die Auswahl von Fontattributen verwenden zu können.

7.13 Paket shortvrb

Dieses kleine Paket ist ein Nebenprodukt des Pakets `doc`, das beim Formatieren der Dokumentation von Standardpaketen eingesetzt wird. Bei diesen Dokumentationen sind sehr häufig im Text $\text{\LaTeX}$ -Befehle zu setzen, und es ist lästig, sie ständig mit dem Befehl `\verb` auszeichnen zu müssen. Aus diesem Grund hat Frank Mittelbach, Autor der beiden Pakete, eine Möglichkeit entwickelt, die Befehle `\verb` und `\verb*` abzukürzen. Da diese Möglichkeit auch an anderen Stellen von Interesse ist, wurde dieser spezielle Teil des `doc`-Pakets als eigenes Paket `shortvrb` zur Verfügung gestellt.

Da Optionen für das Paket nicht vorgesehen sind, wird es einfach mit

```
\usepackage{shortvrb}
```

geladen. Danach kann man Kurzformen für die Befehle `\verb` und `\verb*` definieren.

Die Kurzform eines `\verb`- oder `\verb*`-Befehls bewirkt, dass man den Befehl selbst weglassen kann und ein zu definierendes Zeichen verwendet, das den in `verb` zu setzenden Text markiert. Man legt daher ein Begrenzungszeichen für den gesamten Text fest – wenn beide Befehle benutzt werden unterschiedlich für `\verb` und `\verb*` – und erklärt die Begrenzungszeichen zur Kurzform. Anders ausgedrückt: Die Begrenzungen werden *Aktive Zeichen*.

Zur Definition der als Kurzbefehl zu verwendenden Markierungszeichen stellt das Paket drei Befehle zur Verfügung:

`\MakeShortVerb{Zeichen}`

Macht `<Zeichen>` zum Markierungszeichen von `\verb`-Befehlen, so dass an Stelle von `\verb<Zeichen><Text><Zeichen>` einfach `<Zeichen><Text><Zeichen>` verwendet werden kann.

`\MakeShortVerb*{Zeichen}`

Macht `<Zeichen>` zum Markierungszeichen von `\verb*`-Befehlen und erlaubt hier die Kurzform `<Zeichen><Text><Zeichen>` eines `\verb*`-Befehls.

`\DeleteShortVerb{Zeichen}`

Beendet die Bedeutung von `{Zeichen}` als Kurzform für `\verb` oder `\verb*`, falls ihm diese Bedeutung zugewiesen wurde (andernfalls kein Fehler).

Diese Befehle können erst nach `\begin{document}` verwendet werden.

`<Zeichen>` ist sorgfältig auszuwählen, wenn es als Kurzform verwendet werden soll, da es im gesamten Dokument als Zeichen höchstens in einer `verbatim`-Umgebung vorkommen darf (oder entsprechend häufig gelöscht und neu vereinbart werden muss). In der Regel sollte ein Sonderzeichen wie `|`, `@` oder ähnliches verwendet werden. Von der Verwendung anderer Zeichen wie Buchstaben und Ziffern wird dringend abgeraten. Auch Zeichen mit Befehlscharakter wie `\` oder `"` bei deutscher Sprachanpassung, können nicht verwendet werden.

Zu beachten ist außerdem, dass die Kurzform ebenso wie `\verb` selbst nicht im Argument eines Befehls vorkommen darf. Das hat die Konsequenz, dass auch das für die Kurzform verwendete Zeichen nicht im Argument eines Befehls erlaubt ist, solange es diese Bedeutung hat.

```

7 \usepackage{shortvrb}
8 \begin{document}
9 \MakeShortVerb{|}
10 \MakeShortVerb*{@}
11 \verb.\itshape. und
12 |\itshape| erzeugen die
13 gleiche Ausgabe, ebenso
14 wie \verb*.der Text.
15 und @der Text@.
16 \end{document}

```

`\itshape` und `\itshape` erzeugen die gleiche Ausgabe, ebenso wie `der\_Text` und `der\_Text`.

[Buch1/Basispakete/shortvrb.tex]

7.14 Paket syntonly

In Ausnahmefällen möchte man lediglich die Korrektheit eines $\text{\LaTeX}$ -Dokuments überprüfen, ohne das Dokument tatsächlich zu setzen und eine möglicherweise bereits existierende `.dvi`-Datei zu überschreiben. Bei der Geschwindigkeit der heutigen Rechner spielt die zum Setzen erforderliche Zeit dabei wohl keine Rolle. Es können aber Gründe vorliegen, eine bereits vorhandene `.dvi`-Datei nicht zu überschreiben, etwa bei einer Suche nach einem fatalen Fehler, der die Formatierung abbricht.

In diesem Fall kann das Paket `syntonly` von Frank Mittelbach und Dr. Rainer Schöpf helfen. Es wird geladen mit

```
\usepackage{syntonly}
```

und stellt dem Anwender den Befehl

```
\syntonly
```

zur Verfügung, der *in der Präambel* angegeben werden muss, um beim Formatieren die Ausgabe zu unterdrücken; eine bereits vorhandene `.dvi`-Datei bleibt also erhalten. Ohne diesen Befehl, wird das Dokument normal formatiert, auch wenn das Paket `syntonly` geladen ist.

7.15 Paket textcomp

Bei der Entwicklung der DC- und der daraus weiterentwickelten EC-Fontfamilie wurden viele Buchstaben aus anderen Sprachen in die Fonts

aufgenommen. Trotz Verdoppelung von 128 auf 256 Zeichen pro Font fehlte der Platz für einige der in den 128-Zeichen-Fonts vorhandenen Symbole. So fiel die Entscheidung, jedem Textfont einen Symbolfont zuzuordnen. Der in den Symbolfonts verfügbare Platz wurde genutzt, um viele neue Textsymbole zur Verfügung zu stellen.

Um Anwendern die Verwendung der Textsymbole zu erleichtern, haben die Autoren Johannes Braams, David Carlisle, Alan Jeffrey, Frank Mittelbach, Chris Rowley und i Dr. Rainer Schöpf des fontenc-Pakets das Paket textcomp zur Verfügung gestellt. Es wird geladen mit

```
\usepackage[<Umfang>]{textcomp}
```

und kann auch mit anderen Fontfamilien – beispielsweise mit PostScript-Fonts – verwendet werden. Bei Verwendung anderer Fonts als denen der EC-Fontfamilie für die Symbole sind allerdings in der Regel nicht alle Symbole verfügbar.

Aus diesem Grund kann man durch Angabe der Option *<Umfang>* einstellen, welche Teilmenge der Symbole verfügbar gemacht werden soll. Die folgenden Option können angegeben werden:

- safe** Nur Symbole, die auch im ISO-Adobe-Zeichensatz vorhanden sind. Das Symbol für ‚Währung‘, das häufig für den Euro missbraucht wird.
Alte Type1-Fonts enthalten typischerweise nur diese Teilmenge.
- euro** Wie *safe* aber mit dem Euro-Symbol `\texteuro`.
Die meisten neueren Fonts enthalten diese Teilmenge.
- full** Aktiviert Befehl für alle Zeichen in den EC-Symbolfonts (in der TS1-Kodierung) und ist nur bei den EC- und CM-Fonts angebracht.
- almostfull**
Wie *full*, aber die Befehle `\textcircled` und `\t` werden *nicht* redefiniert, so dass Befehle wie `\copyright` weiter verwendet werden können.
- force** Alle font- und kodierungsspezifischen zusätzlichen Einschränkungen der als weitere Option angegebenen Teilmenge, werden sowohl im Paket als auch in der Konfigurationsdatei `textcomp.cfg` ignoriert, was gefährlich und nur selten nützlich ist.

In der Regel ist die Angabe einer Option allerdings nicht erforderlich, da textcomp für eine viele Fontfamilien die einzusetzende Teilmenge und eventuelle font- und kodierungsspezifische Einschränkungen kennt.

```

1 \documentclass{article}
2 \usepackage[TS1,T1]{fontenc}
3 \usepackage[full]{textcomp}
4 \setlength{\textwidth}{5cm}
5 \setlength{\parindent}{0pt}
6 \pagestyle{empty}
7 \begin{document}
8 30~\textcelsius\
9 3~\texteuro{} und
10 25~\textcent\
11 \texttrademark{}
12 liegt nicht auf
13 diesem Beispiel
14 \end{document}

```

30 °C
 3 € und 25 ¢
 ™ liegt nicht auf diesem Beispiel

[Buch1/Basispakete/textcomp-bsp.tex]

Die Symbolfonts enthalten auch die Ziffern in der alten Form. Daher definiert das Paket den Befehl

```
\oldstylenums{<Zahl>}
```

um Zahlen mit diesen Ziffern zu setzen. Im folgenden Beispiel wird eine Jahreszahl mit den üblichen und mit den alten Ziffern dargestellt.

```

7 \usepackage[full]{textcomp}
8 \begin{document}
9 01234567890\
10 \oldstylenums{01234567890}
11
12 1849--- \oldstylenums{1849}
13 \end{document}

```

01234567890

01234567890

1849 – 1849

[Buch1/Basispakete/textcomp-od.tex]

Für die Textsymbole definiert das Paket eine große Anzahl von Befehlen, mit denen sie gesetzt werden können. Da es sich um viele einfache Befehle ohne Argumente handelt, werden sie der Einfachheit halber in Form einer Tabelle vorgestellt, die das Symbol zusammen mit dem Befehl zum Setzen zeigt.

Akzente

ˆ	\capitalcedilla	€	\capitalogonek
˘	\capitalgrave	˘	\capitalbreve
˙	\capitalacute	·	\capitaldotaccent
ˆ	\capitalcircumflex	˜	\capitaltilde
¨	\capitaldieresis	˘	\capitalhungarumlaut
°	\capitalring	ˇ	\capitalcaron
–	\capitalmacron		

Sonderzeichen

,	<code>\textquotestraightbase</code>	"	<code>\textquotestraightdblbase</code>
—	<code>\texttwelveudash</code>	—	<code>\textthreequartersemdash</code>
\$	<code>\textdollar</code>	'	<code>\textquotesingle</code>
*	<code>\textasteriskcentered</code>	/	<code>\textfractionsolidus</code>
—	<code>\textminus</code>	[[	<code>\textlbrackdbl</code>
]]	<code>\textrbrackdbl</code>	`	<code>\textasciigrave</code>
~	<code>\texttildelow</code>	˘	<code>\textasciibreve</code>
ˆ	<code>\textasciicaron</code>	“	<code>\textgravedbl</code>
”	<code>\textacutedbl</code>	†	<code>\textdagger</code>
‡	<code>\textdaggerdbl</code>		<code>\textbardbl</code>
‰	<code>\textperthousand</code>	•	<code>\textbullet</code>
°C	<code>\textcelsius</code>	f	<code>\textflorin</code>
™	<code>\texttrademark</code>	¢	<code>\textcent</code>
£	<code>\textsterling</code>	¥	<code>\textyen</code>
	<code>\textbrokenbar</code>	§	<code>\textsection</code>
¨	<code>\textasciidieresis</code>	©	<code>\textcopyright</code>
ª	<code>\textordfeminine</code>	¬	<code>\textlnot</code>
®	<code>\textregistered</code>	—	<code>\textasciimacron</code>
°	<code>\textdegree</code>	±	<code>\textpm</code>
²	<code>\texttwosuperior</code>	³	<code>\textthreesuperior</code>
´	<code>\textasciiacute</code>	µ	<code>\textmu</code>
¶	<code>\textparagraph</code>	·	<code>\textperiodcentered</code>
¹	<code>\textonesuperior</code>	º	<code>\textordmasculine</code>
¼	<code>\textonequarter</code>	½	<code>\textonehalf</code>
¾	<code>\textthreequarters</code>	×	<code>\texttimes</code>
÷	<code>\textdiv</code>		

Symbole

€	<code>\texteuro</code>	Ω	<code>\textohm</code>
€	<code>\textestimated</code>	¤	<code>\textcurrency</code>
ˆ	<code>\capitaltie</code>	ˆ	<code>\newtie</code>
ˆ	<code>\capitalnewtie</code>	←	<code>\textleftarrow</code>
→	<code>\textrightarrow</code>	␣	<code>\textblank</code>
=	<code>\textdblhyphen</code>	0	<code>\textzerooldstyle</code>
1	<code>\textoneoldstyle</code>	2	<code>\texttwooldstyle</code>
3	<code>\textthreeoldstyle</code>	4	<code>\textfouroldstyle</code>
5	<code>\textfiveoldstyle</code>	6	<code>\textsixoldstyle</code>
7	<code>\textsevenoldstyle</code>	8	<code>\texteightoldstyle</code>
9	<code>\textnineoldstyle</code>	<	<code>\textlangle</code>
>	<code>\textrangle</code>	U	<code>\textmho</code>

Symbole (Forts.)

○	<code>\textbigcircle</code>	↑	<code>\textuparrow</code>
↓	<code>\textdownarrow</code>	★	<code>\textborn</code>
o o	<code>\textdivorced</code>	†	<code>\textdied</code>
🍃	<code>\textleaf</code>	∞	<code>\textmarried</code>
♪	<code>\textmusicalnote</code>	=	<code>\textdblhyphenchar</code>
\$	<code>\textdollaroldstyle</code>	¢	<code>\textcentoldstyle</code>
⌠	<code>\textcolonmonetary</code>	₩	<code>\textwon</code>
₤	<code>\textnaira</code>	Ⓖ	<code>\textguarani</code>
₱	<code>\textpeso</code>	£	<code>\textlira</code>
℞	<code>\textrecipe</code>	‡	<code>\textinterrobang</code>
⤵	<code>\textinterrobangdown</code>	đ	<code>\textdong</code>
‰	<code>\textpertenthousand</code>	‡	<code>\textpilcrow</code>
฿	<code>\textbaht</code>	№	<code>\textnumero</code>
%	<code>\textdiscount</code>	○	<code>\textopenbullet</code>
SM	<code>\textservicemark</code>	{	<code>\textlquill</code>
}	<code>\textrquill</code>	Ⓢ	<code>\textcopyleft</code>
Ⓟ	<code>\textcircledP</code>	※	<code>\textreferencemark</code>
√	<code>\textsurd</code>	○	<code>\textcircled</code>
˘	<code>\t</code>		

7.16 Paket tracefnt

Gelegentlich möchte genauer über die Verwendung der Fonts in einem $\text{\LaTeX}$ -Dokument informiert werden. Normalerweise gibt das NFSS (New Font Selection Scheme) nur beim Laden eines Fonts Informationen über den betreffenden Font in die `.log`-Datei aus. Anhand dieser Informationen kann man beispielsweise feststellen, ob alle verwendeten Fonts verfügbar waren und welche Fonts gegebenenfalls durch andere ersetzt werden mussten. Über Fontwechsel wird man nicht informiert.

Das Paket `tracefnt` von Frank Mittelbach und Dr. Rainer Schöpf ermöglicht es, weitere Informationen über die Verwendung der Fonts in einem Dokument zu erhalten. Beim Laden mit

```
\usepackage[<Steuerung>]{tracefnt}
```

lässt sich über das optionale Argument `<Steuerung>` der Umfang der angegebenen Informationen und das Verhalten steuern. Dazu stehen die folgenden Optionen zur Verfügung:

errorshow Ausgabe aller Informationen über das Laden von Fonts nur in das `.log`-File. Nur Fehler werden am Bildschirm

angezeigt, nicht aber beispielsweise Angaben über das (erfolgreiche) Ersetzen eines fehlenden Fonts.

warningshow Abgesehen von Fehlern werden auch alle Warnungen am Bildschirm ausgegeben.

Diese Option entspricht genau dem standardmäßigen Verhalten des NFSS, auch wenn tracefmt nicht geladen wurde.

infoshow Zusätzlich zu den Warnungen werden auch die informativen Meldungen am Bildschirm angezeigt, die andernfalls nur in das `.log`-File geschrieben werden.

Diese Option ist voreingestellt, wenn tracefmt geladen wird.

loading Beim Laden eines externen Fonts wird zusätzlich der Fontname ausgegeben. Dies ist allerdings auf Fonts beschränkt, die erst nach dem Laden von tracefmt geladen werden. Das heisst, dass Fontnamen des $\text{\LaTeX}$ -Formats bzw. der Dokumentklasse nicht angezeigt werden können.

debugshow Neben den Informationen über das Laden von Fonts werden auch Informationen über Fontwechsel angezeigt und in die `.log`-Datei ausgegeben. Namen externer Fonts werden wie bei `loading` angezeigt.

Da zu jedem Fontwechsel im Textmodus und – soweit möglich – im mathematischen Modus Informationen ausgegeben werden, ist bei größeren Dokumenten gegebenenfalls die Ausgabe sehr umfangreich.

pausing Alle Fontwarnungen werden wie Fehler behandelt und unterbrechen damit das Formatieren.

Die Optionen `errorshow`, `warningshow` und `infoshow` schreiben die gleichen Informationen in das `.log`-File. Die Option `loading` gibt zusätzlich die Fontnamen aus, und `debugshow` darüberhinaus Informationen über Fontwechsel.

Das Paket enthält keine dem Anwender zugänglichen Definitionen. Nur die Optionen erlauben dem Anwender, das Verhalten zu beeinflussen.

Teil II

Erweiterungen

Sprachunterstützung mit babel

Dr. Leslie Lamport entwickelte $\LaTeX$ ursprünglich zum Schreiben englischsprachiger Texte. Gewisse Ausdrücke wie ‚Table of Contents‘ für das Inhaltsverzeichnis waren intern fest verankert. Mit der internationalen Verbreitung von $\LaTeX$ wurde es notwendig, auch andere Sprachen zu unterstützen. Eine Anpassung an andere Sprachen erforderte zunächst Eingriffe in $\LaTeX$ selbst.

Sehr bald wurden die fest vorgegebenen englischen Bezeichnungen in $\LaTeX$ durch Makros ersetzt, und Autoren aus verschiedenen Ländern entwickelten Pakete, die in den Makros die Bezeichnungen der nationalen Sprachen ersetzen. Schließlich kam Johannes Braams auf die Idee, die wachsende Anzahl der Pakete zur Anpassung von $\LaTeX$ an unterschiedliche Sprachen, die natürlich unheitlich zu bedienen waren, zu vereinheitlichen und in einem einzigen Paket zusammenzufassen.

Neben den Bezeichnungen ist die Worttrennung abhängig von der Sprache. $\TeX$ verwendet für die Trennung Trennmuster, die bei der Generierung von Formaten geladen werden müssen (diese werden in der Regel bei der Installation der $\TeX$ -Software erzeugt). Mit heutigen $\TeX$ -Versionen können eine Reihe unterschiedlicher Trennmuster gleichzeitig geladen werden, so dass ein $\LaTeX$ -Format in der Regel Trennmuster für alle von einem Benutzer benötigten Sprachen enthält.

Eine Rechtschreibprüfung wird von $\TeX$ selbst nicht durchgeführt. Es gibt allerdings mehrere frei verfügbare Programme – beispielsweise `ispell` – die diese Aufgabe übernehmen können. Diese sind in verschiedenen $\TeX$ -Installationen integriert. Wenn hier vor allem im Zusammenhang mit der deutschen Sprache von (alter und neuer) ‚Rechtschreibung‘ die Rede ist, sind damit also nur die zugehörigen Trennregeln gemeint.

Über die genannten Bezeichnungen und Worttrennung hinaus unterscheiden sich verschiedene Sprachräume auch in den Konventionen bei der Angabe eines Datums. Zum einen unterscheiden sich die Namen der Monate, gelegentlich sogar schon bei Dialekten. Auf der anderen Seite gibt es *landesspezifische* Unterschiede selbst bei gleicher Sprache.

Um diese Sprachanpassungen – soweit $\TeX$ sie erfüllen kann – kümmert sich das Paket babel:

1. Aktivieren der Trennmuster für die verlangte(n) Sprache(n);
2. Umstellen der Bezeichnungen auf die aktuelle Sprache oder den Dialekt;
3. Anpassung des Datums an die Konventionen der aktuellen Sprache, des Dialekts oder der Variante.

Zu beachten ist, dass babel Trennmuster nicht laden kann, da $\TeX$ das Laden von Trennmustern nur während der Generierung eines Formats zulässt. babel kann daher nur die Trennmuster aktivieren, die beim Erzeugen des $\LaTeX$ -Formats geladen wurden. Fehlen die Trennmuster zu einer Sprache, aktiviert babel die voreingestellten Trennmuster (in der Regel für Englisch) und gibt eine Warnung aus. In diesem Fall ist damit zu rechnen, dass $\TeX$ nicht korrekt trennen kann.

babel unterstützt eine Vielzahl von Sprachen, Dialekten und Sprachvarianten. Bei der deutschen Sprache gibt es nicht nur Varianten für die Trennung nach den neuen und alten Rechtschreibregeln, sondern auch für in Deutschland und Österreich gebräuchliche Sprachvarianten: Beispielsweise lautet die Monatsbezeichnung in Deutschland *Januar* und *Jänner* in Österreich. Bei den anderen Sprachen sind häufig unterschiedliche Konventionen bei der Angabe des Datums Anlass zur Unterscheidung mehrerer Dialekte.

Die Sprachanpassungen selbst sind Beiträge vieler Autoren. Die bereits unterstützten Sprachen, Dialekte und Varianten sowie die Optionen, über die sie aktiviert werden können, zeigt Tabelle 8.1. Anpassungen an weitere Sprachen sind in Entwicklung, und andere werden folgen.

Mit den Befehlen in babel kann jederzeit eines der bei der Formatgenerierung geladenen Trennmuster aktiviert werden. Zu jedem Zeitpunkt ist genau ein Trennmuster aktiv. Gleichzeitig werden die Bezeichnungen an die aktuell gewählte Sprache angepasst.

Das Paket wird mit dem Befehl

```
\usepackage[<Sprache>[, <Weitere Sprache(n)>]]{babel}
```

geladen. Die Trennmuster für die angegebenen Sprachen werden zugänglich gemacht, wenn das $\LaTeX$ -Format sie zur Verfügung stellt. Fehlen für eine Sprache die Trennmuster, werden für sie die voreingestellten (in der Regel die für Englisch) verwendet. Zugleich aktiviert der Befehl die zuletzt angegebene Sprache und setzt die für sie festgelegten Bezeichnungen ein. Im Dokument kann anschließend zwischen den angegebenen Sprachen gewechselt werden. Tabelle 8.1 enthält die Liste der zum Zeitpunkt der Entstehung dieses Buchs verfügbaren Sprachen und der Optionen, die für sie vorgesehen sind. Mehrere angegebene Optionen zu einer Sprache erlauben eine Auswahl des gewünschten Dialekts oder von lokalen Anpassungen.

Sprache	Option zum Aktivieren der Trennmuster
Afrikaans	afrikaans
Bahasa	bahasa
Baskisch	basque
Bretonisch	breton
Bulgarisch	bulgarian
Dänisch	danish
Deutsch alte Rechtschreibung	german, germanb, austrian)
Deutsch neue Rechtschreibung	ngerman, naustrian)
Englisch	english, american, australian, british, canadian, newzealand, UKenglish, USenglish
Esperanto	esperanto
Estnisch	estonian
Finnisch	finnish
Französisch	french, francais, frenchb, canadian acadian
Galizisch	galician
Griechisch	greek, polutonikogreek
Hebräisch	hebrew
Holländisch	dutch
Interlingua	interlingua
Irisches Gälisch	irish
Isländisch	icelandic
Italienisch	italian
Katalanisch	catalan
Kroatisch	croatian
Latein	latin
Nieder-Sorbisch	lowersorbian
Nord-Samisch	samin
Norwegisch	norsk, nynorsk
Ober-Sorbisch	uppersorbian

Sprache	Option zum Aktivieren der Trennmuster
Polnisch	polish
Portugiesisch	portuges, portuguese, brazilian, brazil
Rumänisch	romanian
Russisch	russian
Schottisches Gälisch	scottish
Schwedisch	swedish
Serbisch	serbian
Slowakisch	slovak
Slowenisch	slovene
Spanisch	spanish
Tschechisch	czech
Türkisch	turkish
Ukrainisch	ukrainian
Ungarisch	hungarian, magyar
Walisisch	welsh

Tabelle 8.1: Sprachen und Optionen, die die Trennmuster der Sprache aktivieren und die Bezeichnungen anpassen

Das Paket babel sollte in der Präambel vor anderen Paketen geladen werden, damit diese Pakete sich auf die gewählte(n) Sprache(n) einstellen können.

Bei einsprachigen Dokumenten ist außer dem Laden des Pakets mit Angabe der zu verwendenden Sprache als Option nichts weiter zu tun, da die Sprache automatisch aktiviert wird. Bei mehrsprachigen Dokumenten ist die *letzte* in der Optionsliste angegebene Sprache aktiv. Die Grundsprache des Dokuments sollte daher als letzte angegeben werden.

Zur Umschaltung zwischen den angeforderten Sprachen, Dialekten und Varianten stellt babel eine Reihe von Befehlen und Umgebungen zur Verfügung. Der zentrale Befehl zur Auswahl der im Folgenden zu verwendenden Sprache ist

```
\selectlanguage{Sprache}
```

mit Angabe einer beim Laden von babel angegebenen *<Sprache>*. Der Befehl stellt alle Spracheinstellungen um: die Trennregeln, sprachspezifische Bezeichnungen, Befehle und das Datum.

Für kurze Textpassagen in einer anderen Sprache kann der Befehl `\foreignlanguage` verwendet werden:

```
\foreignlanguage{<Sprache>}{<Text>}
```

Für *<Text>* werden die Trennregeln und die sprachspezifischen Befehle der angegebenen *<Sprache>* aktiviert. Da es sich nur um kurze Textpassagen handelt, werden die sprachspezifischen Bezeichnungen und die Form des Datums nicht umgesetzt.

Für längere Abschnitte, die in einer anderen Sprache gesetzt werden sollen, ist die Umgebung `otherlanguage` vorgesehen, die auf ihren Inhalt die gleiche Funktion wie ein `\selectlanguage`-Befehl hat. Der Unterschied zu `\selectlanguage` ist lediglich, dass nach dem Ende der Umgebung wieder die vorher aktive Sprache aktiviert wird. Umgebung hat die folgende Form:

```
\begin{otherlanguage}{<Sprache>}
...
\end{otherlanguage}
```

Ist eine Anpassung sprachspezifischer Bezeichnungen nicht gewünscht, kann auch die Form

```
\begin{otherlanguage*}{<Sprache>}
...
\end{otherlanguage*}
```

verwendet werden, die sich ähnlich wie `\foreignlanguage` verhält.

Unabhängig von der Sprache sind Befehle für eine Reihe von Sonderzeichen und Buchstaben vordefiniert. Die folgende Tabelle gibt eine Übersicht über die Befehle und den ihnen zugeordneten Text:

<code>\glq</code>	,	<code>\grq</code>	‘	<code>\glqq</code>	„	<code>\grqq</code>	“
<code>\flq</code>	<	<code>\frq</code>	>	<code>\flqq</code>	«	<code>\frqq</code>	»
<code>\ij</code>	ij	<code>\IJ</code>	IJ	<code>\dj</code>	đ	<code>\DJ</code>	Đ
<code>\ss</code>	ss						

Für die meisten Anführungszeichen sind auch Synonyme definiert. Die folgende Tabelle gibt eine Übersicht.

<code>\guilsinglleft</code>	<code>(\flq)</code>	<	<code>\guilsinglright</code>	<code>(\frq)</code>	>
<code>\guillemotleft</code>	<code>(\flqq)</code>	«	<code>\guillemotright</code>	<code>(\frqq)</code>	»
<code>\quotesinglbase</code>	<code>(\glq)</code>	,	<code>\quotedblbase</code>	<code>(\glqq)</code>	„

Das aktuelle Tagesdatum wird für alle als Optionen angegebenen Sprachen, Dialekte und Sprachvarianten zur Verfügung gestellt. Die einer angegebenen Option *<Sprache>* entsprechende Datumform wird generell mit einem Befehl

```
\date<Sprache>
```

im Befehl `\today` zur Verfügung gestellt.

Eine vollständige Behandlung aller Sprachen und Dialekte würde den Rahmen dieses Buchs sprengen. Betrachtet werden daher nur die für deutschsprachige Leser wohl wichtigsten Sprachen Deutsch, Englisch und Französisch und ihre Dialekte. Einzelheiten zu den übrigen Sprachen und Dialekten sind in der Dokumentation des Pakets zu finden, Braams (2004).

8.1 Texte in deutscher Sprache

Im deutschsprachigen Raum wird unterschieden zwischen den Trennregeln der alten und der neuen sowie zwischen dem Sprachgebrauch in Deutschland und Österreich. Daher stellt `babel` die vier in Tabelle 8.2 angegebenen Varianten zur Verfügung. Die Optionen `german` und `germanb` stehen für die gleiche Variante.

Tabelle 8.2: Sprachoptionen für Deutsch im Paket `babel`

	Deutschland	Österreich
alte Rechtschreibung	german germanb	austrian
neue Rechtschreibung	ngerman	naustrian

Die Varianten für die neue und alte Rechtschreibung unterscheiden sich insbesondere in den verwendeten Trennregeln. Außerdem entfallen bei der neuen Rechtschreibung einige Befehle für Sonderfälle beim Trennen. Die Dialekte für Deutschland und Österreich unterscheiden sich lediglich in einem Monatsnamen.

8.1.1 Gemeinsame Abkürzungen und Befehle

Gemeinsam ist allen Sprachvarianten für Deutsch, dass " ein aktives Zeichen ist wie \. Es wird zur Eingabe der deutschen Umlaute und einiger anderer deutscher Sonderzeichen und Ligaturen verwendet. Allen Varianten gemeinsam sind die folgenden Abkürzungen:

"a, "A, "o, "O, "u, "U
Die Umlaute ä, Ä, ö, Ö, ü, Ü.

"s, "S, "z, "Z
Scharfes s und Varianten: ß, SS, ß, SZ.

"e, "E, "i, "I
Buchstaben mit Trema: ë, Ê, ï, Î.

"\..."' , "<...">
Anführungszeichen: „...“, «...».

"- Einfügen einer Trennstelle in einem Wort, wobei im Gegensatz zu \- die Wortteile links und rechts von der Trennstelle nach den Trennregeln aus den Trennmustern getrennt werden können.

"" Wie "-", wobei jedoch *kein* Bindestrich im Fall einer Trennung an dieser Stelle ausgegeben wird.

"| Verhindert eine Ligatur an dieser Stelle.

"~ Ein Bindestrich, bei dem nicht getrennt werden darf, wie bei ,berg-auf und -ab', ,Ein-/Ausgang' oder ,Primär-(Haupt-)Strom'.

"= Ein Bindestrich, bei dem Trennung erlaubt ist. Die durch diesen Bindestrich getrennten Wortbestandteile können weiterhin automatisch getrennt werden.

```

8  "Umlaute"' :
9  "a, "O, "u, "A, "O, "U, "s\
10 [\smallskipamount]
11 Superkali"-fragilistik%
12 expialigetisch
13 oder hei"st es doch
14 "~gorisch?\\
15 Bei Auf"|lage muss die
16 Ligatur fl getrennt
17 werden.\\
18 Beim Prim"ar-(Haupt"~)"Strom
19 die Trennung beim Bindestrich
20 vor der Klammer verhindern
21 und danach wieder erlauben.
```

„Umlaute“: ä, ö, ü, Ä, Ö, Ü, ß
Superkalifragilistikexpialigetisch
oder heißt es doch -gorisch?
Bei Auflage muss die Ligatur fl
getrennt werden.
Beim Primär-(Haupt-)Strom die
Trennung beim Bindestrich vor
der Klammer verhindern und da-
nach wieder erlauben.

Auch die folgenden Befehle sind für alle Dialekte definiert:

- `\dq`Die Anführungsstriche als Zeichen: "
- `\mdqon`Macht " zu einem aktiven Zeichen wie \
- `\mdqoff`Macht " zu einem normalen Sonderzeichen. Die Abkürzungen für die deutschen Sonderzeichen stehen nach diesem Befehl nicht mehr zur Verfügung.

Keine Unterschiede zwischen den Dialekten gibt es auch bei den sprachspezifischen Bezeichnungen. Die Tabelle 8.3 zeigt die Befehle und die ihnen zugeordneten deutschen Bezeichnungen:

Befehl	Text
<code>\abstractname</code>	Zusammenfassung
<code>\alsoname</code>	siehe auch
<code>\appendixname</code>	Anhang
<code>\bibname</code>	Literaturverzeichnis
<code>\ccname</code>	Verteiler
<code>\chaptername</code>	Kapitel
<code>\contentsname</code>	Inhaltsverzeichnis
<code>\enclname</code>	Anlage(n)
<code>\figurename</code>	Abbildung
<code>\glossaryname</code>	Glossar
<code>\headtoname</code>	An
<code>\indexname</code>	Index
<code>\listfigurename</code>	Abbildungsverzeichnis
<code>\listtablename</code>	Tabellenverzeichnis
<code>\pagename</code>	Seite
<code>\partname</code>	Teil
<code>\prefacename</code>	Vorwort
<code>\proofname</code>	Beweis
<code>\refname</code>	Literatur
<code>\seename</code>	siehe
<code>\tablename</code>	Tabelle

Tabelle 8.3: Bezeichnungen für Deutsch

Gemeinsam sind allen fünf Varianten auch die Monatsnamen für die Monate Februar bis Dezember. Der erste Monat des Jahres wird in den deut-

schen Sprachvarianten als Januar und in den österreichischen als Jänner gesetzt:

Januar ^a Jänner ^b	April	Juli	Oktober
Februar	Mai	August	November
März	Juni	September	Dezember

^a Für die Optionen `german` und `ngerman`

^b Für die Optionen `austrian` und `naustrian`

Beim Datum gibt es außer den eben genannten keine Unterschiede. Die beiden Hauptvarianten sind

```
\dategerman    Deutschlanda
\dateaustrian  Österreichb
```

^a Optionen `german`, `germanb`, `ngerman`

^b Optionen `austrian`, `naustrian`

und definieren das Tagesdatum `\today` wie folgt:

```
<Tag>.\~<Monatsname> <Jahr>
```

Beispiel:

```
12.\~Februar 2006
```

Einziger Unterschied zwischen beiden Befehlen ist der Monatsname *Januar* im Fall von `\dategerman` und *Jänner* bei `\dateaustrian` für den ersten Monat des Jahres. `<Tag>` und `<Jahr>` werden als Zahlen angegeben, das Jahr vierstellig.

8.1.2 Zusätze bei der alten Rechtschreibung

Die Sprachvarianten `german` und `austrian` aktivieren die Trennmuster der alten Rechtschreibung und definieren Abkürzungen und einen Befehl, die bei der neuen Rechtschreibung nicht mehr benötigt werden. Die folgenden Abkürzungen stehen zusätzlich zur Verfügung:

"ck, "CK

Die Buchstaben `ck` und `CK`, die beim Trennen direkt danach in `k-k` oder `K-K` umgewandelt werden: `"ck` → `k-k`, `"CK` → `K-K`.

"f, "F, "l, "L, "m, "M, "n, "N, "p, "P, "r, "R, "t, "T
 Die Konsonanten "f, "F, "l, "L, "m, "M, "n, "N, "p, "P, "r, "R, "t und
 "T, die beim Trennen direkt danach verdoppelt werden: "ff → f-
 f, "FF → F-F, "ll → l-l, "LL → L-L, "mm → m-m, "MM → M-
 M, "nn → n-n, "NN → N-N, "pp → p-p, "PP → P-P, "rr → r-r,
 "RR → R-R, "tt → t-t, "TT → T-T.

Als Alternative zu "ck kann in den Varianten der alten Rechtschreibung der Befehl

```
\ck
```

verwendet werden, der die Buchstabenfolge ck und bei Trennung k-k erzeugt.

8.1.3 Kleines Beispiel

Im folgenden Beispiel soll zum einen die Anwendung von babel demonstriert werden. Zum anderen ist dieses Beispiel aber bewusst mit dem KOMA-Script-Paket (wird in Band 2 erläutert) und der PostScript-Schriftart ,bookman' (vgl. 11.2, S. 351) gesetzt, um alternative Layouts zu zeigen.

```
1 \documentclass[12pt,a4paper,halbparskip*]{scrartcl}
2 \usepackage[ngerman,english]{babel}
3 \usepackage[T1]{fontenc}
4 \usepackage{bookman}
5 \setlength{\textwidth}{11cm} % nur fuer Buch
6 \small % nur für Buch
7
8 \begin{document}
9 \selectlanguage{ngerman}
10 \title{Artikel f"ur keine Zeitschrift}
11 \author{Anna Musterfrau}
12 \maketitle
13 \tableofcontents
14 \thispagestyle{empty} % nur fuer Buch
15 \section{Erstes Kapitel}
16 Und nun ein wenig seri"oser Text, der m"oglichst
17 viele Umlaute enth"alt.
18 Nat"urlich muss auch ein Zitat her
19 "'Kreative Menschen brauchen sinnfreie R"aume"'
20 (Quelle: unbekannt).
21 \section{Weiteres Kapitel}
22 Weiteren Text zu schreiben f"allt schwer.
23 \end{document}
```

Artikel für keine Zeitschrift

Anna Musterfrau

26. Februar 2006

Inhaltsverzeichnis

1	Erstes Kapitel	1
2	Weiteres Kapitel	1

1 Erstes Kapitel

Und nun ein wenig seriöser Text, der möglichst viele Umlaute enthält. Natürlich muss auch ein Zitat her „Kreative Menschen brauchen sinnfreie Räume“ (Quelle: unbekannt).

2 Weiteres Kapitel

Weiteren Text zu schreiben fällt schwer.

8.2 Englische Texte

Englisch ist sozusagen die ‚Muttersprache‘ von $\text{\TeX}$. Daher werden für Englisch keine Befehle für spezielle Zeichen oder Abkürzungen benötigt. Auch die Bezeichnungen stimmen überein. Die Sprachvarianten haben daher lediglich die Aufgabe, das Datum an unterschiedliche nationale Konventionen anzupassen. Tabelle 8.4 gibt eine Übersicht über die Optionen für englische Texte. Die Bezeichnungen sind für alle englischen

Tabelle 8.4: Sprachoptionen für Englisch im Paket babel

Land	Option(en)
USA	english, american, USenglish
Kanada	canadian
Großbritannien	british, UKenglish
Australien	australian
Neuseeland	newzealand

Varianten gleich. Tabelle 8.5 gibt eine Übersicht über die englischen Bezeichnungen.

Auch bei den Monatsnamen unterscheiden sich die englischen Sprachvarianten nicht. Die Monatsnamen lauten:

January	April	July	October
February	May	August	November
March	June	September	December

Einziger Unterschied zwischen den Sprachvarianten sind die Konventionen bei Datumsangaben. Unterschieden werden die drei Konventionen für Großbritannien, Nordamerika und Australien/Neuseeland.

<code>\dateenglish</code>	USA und Kanada ^a
<code>\datebritish</code>	Großbritannien ^b
<code>\dateaustralian</code>	Australien und Neuseeland ^c

^a Optionen english, american, canadian, USenglish

^b Optionen british, UKenglish

^c Optionen australian, newzealand

Mit den oben angegebenen Monatsnamen hat das Tagesdatum `\today` die folgende Form:

<code><Monatsname>~<Tag></code> ,	<code><Jahr></code>	USA und Kanada
<code><TagO>~<Monatsname></code>	<code><Jahr></code>	Großbritannien
<code><Tag>~<Monatsname></code>	<code><Jahr></code>	Australien und Neuseeland

`<Tag>` und `<Jahr>` werden als Zahlen angegeben, das Jahr vierstellig. In Großbritannien wird der Tag `<TagO>` als Ordnungszahl angegeben, und zwar als Zahl mit angehängter Endung `st`, `nd`, `rd` oder `th`, je nachdem, welche Ordnungszahl ausgesprochen wird.

Befehl	Text
<code>\abstractname</code>	Abstract
<code>\alsoname</code>	see also
<code>\appendixname</code>	Appendix
<code>\bibname</code>	Bibliography
<code>\ccname</code>	cc
<code>\chaptername</code>	Chapter
<code>\contentsname</code>	Contents
<code>\enclname</code>	encl
<code>\figurename</code>	Figure
<code>\glossaryname</code>	Glossary
<code>\headtoname</code>	To
<code>\indexname</code>	Index
<code>\listfigurename</code>	List of Figures
<code>\listtablename</code>	List of Tables
<code>\pagename</code>	Page
<code>\partname</code>	Part
<code>\prefacename</code>	Preface
<code>\proofname</code>	Proof
<code>\refname</code>	References
<code>\seename</code>	see
<code>\tablename</code>	Table

Tabelle 8.5: Bezeichnungen für Englisch

8.3 Französische Texte

Die französischen Konventionen unterscheiden sich in einigen Bereichen von den englischen und deutschen. Das betrifft etwa das Einrücken des

ersten Absatzes nach einer Überschrift oder Abstände vor und nach einigen Satzzeichen. Auch die Markierungen und Abstände in Listen weichen üblicherweise vom $\LaTeX$ -Layout ab. Beim Format von Zahldarstellungen gibt es ebenfalls Unterschiede. babel setzt die erwähnten Konventionen und eine Reihe weiterer automatisch um oder stellt Befehle zur Verfügung, die beispielsweise Zahlen in französischer Formatierung erzeugen. Zusätzlich werden Befehle angeboten, die die automatisch berücksichtigten französischen Konventionen (vorübergehend) deaktivieren können. Dies führt dazu, dass babel für die französischen Sprachoptionen einen umfangreichen Befehlssatz zur Verfügung stellt.

Die folgende Liste gibt eine Übersicht über einige Abweichungen der französischen Konventionen vom $\LaTeX$ -Standard, die von den französischen Sprachoptionen unterstützt werden.

- Der erste Absatz nach einer Überschrift wird eingerückt.
- In `itemize`-Listen werden die Einträge auf allen Ebenen mit ‚`–`‘ markiert, und die Abstände zwischen den Einträgen entfallen.
- In anderen Listen werden die Abstände zwischen den Einträgen verringert.
- Bei einigen Satzzeichen ist ein Abstand *vor* dem Satzzeichen üblich.
- Nach `\dots` wird kein Abstand gelassen.

Nach dem Wechsel in die französische Sprache beispielsweise mit `\selectlanguage{french}` werden diese Konventionen berücksichtigt, solange die Sprache nicht erneut gewechselt wird. Die französischen Sprachoptionen stellen eine Reihe von Befehlen zur Verfügung, um gegebenenfalls einzelne dieser Konventionen zu missachten und durch die $\LaTeX$ -Konventionen zu ersetzen.

Um die mit den Befehl vorgenommenen Änderungen nicht in jedem Dokument neu vornehmen zu müssen, ist eine Konfigurationsdatei unter dem Namen `frenchb.cfg` vorgesehen. Die in der Distribution enthaltene Konfigurationsdatei besteht nur aus Kommentaren, die die (nicht aktivierten) Befehle beschreiben. Sie sollte gegebenenfalls in ein eigenes Verzeichnis kopiert und dort angepasst werden. Die meisten dort erwähnten Befehle werden im Folgenden beschrieben.

Die französischen Sprachoptionen und ihre Einsatzbereiche zeigt Tabelle 8.6. Die Option `frenchb` und `français` dienen der Kompatibilität mit älteren Dokumenten und sollten in neuen Dokumenten nicht mehr verwendet werden.

Um die nach den französischen Konventionen vom $\LaTeX$ -Standard abweichenden Abstände bei einigen Satzzeichen korrekt berücksichtigen

Tabelle 8.6: Sprachoptionen für Französisch im Paket babel

Land	Option(en)
Frankreich	french, frenchb, francais
Kanada	canadien, acadian

zu können, werden die betreffenden Satzzeichen zu aktiven Zeichen gemacht. Es handelt sich dabei um die Satzzeichen

; ! ? :

Die Dokumentation empfiehlt, vor diesen vier Satzzeichen entsprechend der französischen Konvention ein Leerzeichen einzugeben, auch wenn es nicht erforderlich ist, um einen Abstand vor einem dieser Satzzeichen zu erhalten. Zu beachten ist, dass diese Satzzeichen *grundsätzlich* in dieser Weise gesetzt werden, wenn die französische Sprache aktiviert ist. Soll eines der Satzzeichen *ohne* den zusätzlichen Abstand gesetzt werden, müssen die Zeichen in den Formen `\string;`, `\string!`, `\string?` oder `\string:` eingegeben werden.

Bei den Bezeichnungen gibt es keine Unterschiede zwischen den französischen Sprachvarianten. Tabelle 8.7 zeigt die französischen Bezeichnungen.

Befehl	Text
<code>\abstractname</code>	Résumé
<code>\alsoname</code>	<i>voir aussi</i>
<code>\appendixname</code>	Annexe
<code>\bibname</code>	Bibliographie
<code>\ccname</code>	Copie à
<code>\chaptername</code>	Chapitre
<code>\contentsname</code>	Table des matières
<code>\enclname</code>	P. J.
<code>\figurename</code>	FIG.
<code>\glossaryname</code>	Glossaire
<code>\headtoname</code>	
<code>\indexname</code>	Index
<code>\listfigurename</code>	Table des figures
<code>\listtablename</code>	Liste des tableaux
<code>\pagename</code>	page
<code>\partname</code>	<Nummer> partie
<code>\prefacename</code>	Préface
<code>\proofname</code>	Démonstration
<code>\refname</code>	Références
<code>\seename</code>	<i>voir</i>

Befehl	Text
<code>\tablename</code>	TAB.

Tabelle 8.7: Bezeichnungen für Französisch

Eine Besonderheit ist `\partname`. Dieser Befehl setzt bei den französischen Sprachvarianten nicht nur die Bezeichnung, sondern auch die Nummer des Teils. Diese spezielle Konstruktion ist erforderlich, um die französische Konvention, die Nummer der Teile nicht als Zahl sondern textuell anzugeben. `<Nummer>` ist daher die ausgeschriebene Ordnungszahl bis einschließlich zwanzig, die dem Zähler `part` entspricht: ‚Première‘, ‚Deuxième‘, ‚Troisième‘, ... , ‚Vingtième‘. Bei Querverweisen wird die Nummer des Teils wie üblich mit römischen Ziffern angegeben.

Die Monatsnamen sind bei allen Sprachvarianten gleich. Die folgende Tabelle führt diese Monatsnamen auf.

janvier	avril	juillet	octobre
février	mai	août	novembre
mars	juin	septembre	décembre

Auch das Datum wird bei allen Sprachvarianten nach den gleichen Konventionen gesetzt. Es gibt daher keinen Anlass, das Datumformat beispielsweise mit `\datefrench` umzuschalten. Das Tagesdatum `\today` ist bei allen französischen Sprachvarianten wie folgt definiert:

```
<Tag>~<Monatsname> <Jahr>
```

`<Tag>` und `<Jahr>` werden als Zahlen angegeben, das Jahr vierstellig. Der erste Tag eines Monats wird dadurch ausgezeichnet, dass statt einer einfachen Zahlangabe `<Tag>` die Ordnungszahl in der Form ‚1^{er}‘ (durch angehängtes `\ier`) gesetzt wird.

Die folgende Liste gibt eine Übersicht über die im Benutzerhandbuch von Braams (2004) aufgeführten Befehle. Einige interessante Befehle, die nur in der Programmdokumentation beschrieben sind, wurden ebenfalls (mit einem entsprechenden Hinweis) aufgenommen. Die Liste ist untergliedert in Befehle, die das französische Layout betreffen, und sonstige für französische Texte relevante Befehle.

Layoutanpassungen

`\FrenchLayout` `\StandardLayout`

`\FrenchLayout` aktiviert und `\StandardLayout` deaktiviert das den französischen Konventionen entsprechende Layout unabhängig von der gerade aktiven Sprache bei mehrsprachigen Dokumenten.

Beide Befehle können nur in der Präambel verwendet werden.

`\FrenchFootnotes` `\StandardFootnotes`

Die französischen Norm-Spezifikationen legen fest, dass die Nummer der Fußnoten vor dem Fußnotentext in der Form ,1. ‘ und nicht als ,¹‘ gesetzt werden sollen. `\FrenchFootnotes` aktiviert diese Form der Angabe der Nummern vor den Fußnoten, die nicht automatisch aktiviert ist. Der Satz von Fußnotensymbolen wie etwa bei `\thanks` wird nicht verändert. `\StandardFootnotes` wechselt wieder zur gewohnten Darstellung der Fußnotennummern und kann auch verwendet werden, um lokal in einer `minipage` Fußnoten nach dem $\text{\LaTeX}$ -Standard zu setzen.

Die Umschaltung der Darstellung der Fußnotennummern mit diesen Befehlen erfolgt unabhängig von der gerade aktiven Sprache.

`\AddThinSpaceBeforeFootnotes`

Die französischen Norm-Spezifikationen legen auch fest, dass der Verweis auf Fußnoten im laufenden Text durch einen kleinen Abstand vom Text getrennt werden soll. Dieser Befehl kann *in der Präambel* angegeben werden, um vor Fußnotenmarkierungen einen kleinen Abstand einzufügen, da andernfalls auch in französischen Dokumenten Fußnotenmarkierungen nach dem $\text{\LaTeX}$ -Standard formatiert werden.

Der Abstand wird nach der Aktivierung unabhängig von der gerade aktiven Sprache eingefügt.

`\FrenchListSpacingfalse`

Nach den französischen Konventionen werden Listeneinträge generell mit verkürztem vertikalem Abstand gesetzt. Mit diesem Befehl lässt sich der (für $\text{\LaTeX}$) normale vertikale Abstand auch in französischen Textabschnitten wieder herstellen.

`\FrenchItemizeSpacingfalse`

Entsprechend den französischen Konventionen werden `itemize`-Listen ohne vertikalen Abstand zwischen den Listeneinträgen ge-

setzt. Dieser Befehl erlaubt es, auch in französischen Textabschnitten `itemize`-Listen mit den Standardabständen von L^AT_EX zu setzen.

`\FrenchLabelItem`

Nach französischer Konvention werden `itemize`-Listen auf allen vier vorgesehenen Ebenen mit dem Zeichen ‚`‘` (`\FrenchLabelItem`) markiert. Diese Konvention wird in französischen Textabschnitten befolgt. Soll statt dessen ein anderes Zeichen, etwa ‚`—`‘, verwendet werden, kann der Befehl `\FrenchLabelItem` entsprechend redefiniert werden.

Sollen in französischen Textabschnitten die Markierungen für einzelne Ebenen geändert werden, kann das mit einem Befehl

`\renewcommand{item<nn>}{<Marke>}`

geschehen. `<nn>` steht dabei für eine der Ebenen `i`, `ii`, `iii` oder `iv`. So vorgenommene Änderungen werden gespeichert. Nach einem Wechsel in eine andere Sprache werden die Einträge in `itemize`-Listen wieder wie üblich markiert. Nach der Rückkehr zur französischen Sprache werden Änderungen erneut aktiviert.

`\NoAutoSpaceBeforeFDP`

`\AutoSpaceBeforeFDP`

Die aktiven (Satz-)Zeichen `;`, `!`, `?` und `:` fügen entsprechend den französischen Konventionen selbständig einen Abstand vor dem Zeichen ein. Da dieses Verhalten nicht von allen Anwendern bevorzugt wird, kann es mit dem Befehl `\NoAutoSpaceBeforeFDP` abgeschaltet werden. Danach beachtet L^AT_EX exakt die Eingabe und setzt nur dann einen Abstand vor diesen Zeichen, wenn er vom Anwender eingegeben wurde.

`\AutoSpaceBeforeFDP` wechselt wieder zum voreingestellten Verhalten der genannten Satzzeichen und fügt erneut automatisch einen Abstand vor ihnen ein.

Spezielle Zeichen und Konventionen

`\og`

`\fg`

Die französischen Anführungszeichen ‚`«`‘ und ‚`»`‘. Abstände zwischen den Anführungszeichen und dem eingeschlossenen Text werden automatisch eingefügt. Beispielsweise wird `\og{ }Text\fg` als `«Text»` gesetzt.

Statt der beiden Befehle können auch `<<` und `>>` verwendet werden: `<<Text>>` erzeugt «Text». Bei dieser Eingabe werden aber keine Abstände eingefügt.

`\bsc{<Familiennamen>}`

Nach französischen Konventionen werden Familiennamen in der Schriftart ‚Caps and Smallcaps‘ gesetzt. Dieser Befehl dient dazu, einen Familiennamen entsprechend der Konvention zu setzen. Beispielsweise wird Donald E. `\bsc{Knuth}` in der Form ‚Donald E. KNUTH‘ gesetzt.

`\up`

Setzt bei Abkürzungen wie beispielsweise ‚M^{me}‘ für ‚Madame‘ (Eingabe: `M\up{me}`) oder ‚1^{er}‘ (Eingabe: `1\up{er}`) für ‚premier‘ den hochgestellten Teil

`\No`

`\no`

Abkürzungen für ‚Numéro‘ (`\No`) und ‚numéro‘ (`\no`); ‚N^o‘ und ‚n^o‘

`\ieme`

`\iemes`

`\ier`

`\iers`

`\iere`

`\ieres`

Diese Befehle dienen zur Angabe von Ordnungszahlen in Form von Ziffern mit hochgestellter Endung:

<code>2\ieme</code>	2 ^e	<code>3\iemes</code>	3 ^{es}
<code>4\ier</code>	4 ^{er}	<code>5\iers</code>	5 ^{ers}
<code>6\iere</code>	6 ^{re}	<code>7\ieres</code>	7 ^{res}

`\degree`

`\degrees{<Text>}`

`\degree` setzt das Symbol für Grad wie in `75\degree` für ‚75°‘. `\degrees` dient beispielsweise zur Angabe von Temperaturen wie in `29 \degrees{C}` für ‚29 °C‘.

`\nombre[<Dp>]{<Zahl>}`

`\ThinSpaceInFrenchNumbers`

Auch für Zahlen gelten in Französisch besondere Konventionen. Der Befehl `\nombre` erlaubt sowohl im Textmodus als auch im mathematischen Modus die einfache Eingabe der Zahlen und setzt sie gemäß den französischen Konventionen. Beispielsweise wird die Eingabe `\nombre{1234567,3456789}` bei aktiver französischer Sprache in der Form ‚1 234 567,345 678 9‘ gesetzt. In anderen Sprachen wie Deutsch oder Englisch hat die Ausgabe hingegen die Form 1,234,567.345,678,9. Bei der Zahlangabe wird vorausgesetzt, dass die Zahlen wie im Französischen und auch im Deutschen üblich mit einem *Dezimalkomma* angegeben werden. Der Befehl ist unabhängig von der gerade aktiven Sprache. Das optionale

Argument `<Dp>` erlaubt die Angabe eines Zeichens für den ‚Dezimalpunkt‘ beim Setzen der Zahl.

Durch Angabe von `\ThinSpaceInFrenchNumbers` in der Präambel oder in der Konfigurationsdatei lässt sich der Abstand zur Gliederung der Ziffernfolgen vor und nach dem Dezimalkomma verringern.

`\decimalsep` `\thousandsep`

Diese beiden nur in der Programmdokumentation erwähnten Befehle legen fest, welches Zeichen Vor- und Nachkommastellen trennen soll (`\decimalsep`) und welches zur Gliederung der Vor- und Nachkommastellen in Dreiergruppen (`\thousandsep`). Voreingestellt sind Komma oder Punkt für `\decimalsep` sowie ein nicht trennendes Leerzeichen oder ein Komma für `\thousandsep` (jeweils in Französisch oder anderen Sprachen).

`\DecimalMathComma` `\StandardMathComma`

Im mathematischen Modus wird nach einem Komma ein kleiner Abstand eingefügt. Dieser Befehl bewirkt, dass danach, wenn die französische Sprache aktiv ist, dieser Abstand nach einem Komma entfällt. Auf mathematische Formeln in anderen Sprachen hat der Befehl keinen Einfluss.

Der Befehl `\StandardMathComma` legt fest, dass nach einem Komma im mathematischen Modus wieder ein kleiner Abstand eingefügt wird, wie es dem $\text{\LaTeX}$ -Standard entspricht.

`\LasyGuillemets`

`\FrenchGuillemetsFrom{<K>}{<F>}{<L>}{<R>}`

Diese beiden in der Präambel zu verwendenden Befehle dienen dazu, französische Anführungszeichen (‚Guillemets‘) zu emulieren oder aus einem anderen Font zu laden, wenn der verwendete Font – beispielsweise ein OT1-kodierter Font – sie nicht enthält. `\LasyGuillemets` emuliert sie mit Hilfe der Zeichenfonts von $\text{\TeX}$ und kann daher stets verwendet werden. Weitere Angaben sind nicht erforderlich.

Der zweite Befehl dient dazu, die ‚Guillemets‘ aus einem anderen Font zu laden. Dazu sind vier Argumente anzugeben. Das erste Argument `<K>` gibt die $\text{\TeX}$ -Kodierung des zu ladenden Fonts an, etwa T1. Das zweite Argument `<F>` bezeichnet die Fontfamilie (beispielsweise `ppl` für die Schriftart *Palatino*), aus dem die Anführungszeichen genommen werden sollen. Das dritte und vierte Argument geben die Positionen des linken (`<L>`) und rechten (`<R>`)

Anführungszeichens im Font $\langle F \rangle$ unter der Codierung $\langle K \rangle$ an. Um die ‚Guillemets‘ aus dem Font *Palatino* zu verwenden, ist der Befehl

```
\FrenchGuillemetsFrom{T1}{ppl}{19}{20}
```

in der Präambel anzugeben.

```
\primo \secundo \tertio \quarto
```

Für Aufzählungen oder als Marken in (kurzen) Listen sind diese Befehle gedacht. Die Eingabe

```
\primo a; \secundo b; \tertio c; \quarto d
```

erzeugt die Ausgabe 1° a; 2° b; 3° c; 4° d.

```
\fprimo) \fsecundo) \ftertio) \fquarto)
```

Eine Variante der Befehle `\primo, ...`, bei der der Abstand zwischen ‚1°‘ und der folgenden Klammer unterdrückt wird, wie er bei `\fprimo)` auftritt: ‚1°)‘. Die Eingabe

```
\fprimo) a; \fsecundo) b; \ftertio) c;
\fquarto) d
```

wird als

```
1°) a; 2°) b; 3°) c;
4°) d
```

gesetzt.

Diese Befehl sind nicht in Braams (2004) dokumentiert und nur in der Dokumentation des Codes beschrieben.

```
\FrenchEnumerate{<Zahl>}
```

```
\FrenchPopularEnumerate{<Zahl>}
```

Diese Befehl sind allgemeine Formen der Befehle `\primo ...` und `\fprimo)`, bei denen die Zahl selbst als Argument angegeben wird. `\FrenchEnumerate{2}` entspricht beispielsweise `\secundo` und `\FrenchPopularEnumerate{3}` entspricht `\ftertio`. `\FrenchEnumerate{5}` ergibt also ‚5°‘ und `\FrenchPopularEnumerate{10}` ergibt ‚10°‘.

Auch diese Befehl sind nicht in Braams (2004), sondern nur in der Dokumentation des Codes beschrieben.

Schließlich werden im Französischen noch einige spezielle Buchstaben benötigt, und für einige bisher noch nicht erwähnte Sonderzeichen werden spezielle Befehle zur Verfügung gestellt. Die folgende Tabelle gibt eine Übersicht über diese Buchstaben und Sonderzeichen:

<code>\ae</code>	æ	<code>\AE</code>	Æ	<code>\oe</code>	œ	<code>\OE</code>	Œ
<code>\at</code>	@	<code>\tild</code>	~	<code>\boi</code>	\	<code>\circonflexe</code>	^

8.3.1 Vollständiges Beispiel

Das folgende Beispiel ist eine Erweiterung der Vorlage von Seite 214. Die Anwendung von babel für verschiedene Sprachen innerhalb eines Dokuments wird gezeigt. Dieses Beispiel ist mit dem KOMA-Script-Paket (wird in Band 2 erläutert) und der PostScript-Schriftart ‚bookman‘ (vgl. 11.2, S. 351) gesetzt.

Auf das Erzeugen des Artikeltitels wird durch Auskommentieren der entsprechenden Zeile 11 verzichtet.

Die Zeilen 29–32 enthalten englischen Text, die Zeilen 43–47 ein französisches Zitat. In den Zeilen 28, 34, 42 und 49 wird die Sprache entsprechend gewechselt. Wie man sieht kann dann in fremdsprachigen Abschnitten auch die in der jeweiligen Sprache übliche Kodierung für die Umlaute verwendet werden.

```

1  \documentclass[a4paper,halfparskip]{scrartcl}
2  \usepackage[ngerman,english,french]{babel}
3  \usepackage[T1]{fontenc}
4  \usepackage{bookman}
5  \setlength{\textwidth}{11cm} % nur fuer Buch
6
7  \begin{document}
8  \selectlanguage{ngerman}
9  \title{Artikel f"ur keine Zeitschrift}
10 \author{Anna Musterfrau}
11 %\maketitle % auskommentiert
12 \tableofcontents
13 \thispagestyle{empty} % nur fuer Buch
14
15 \section{Erstes Kapitel}
16
17 Und nun ein wenig seri"oser Text, der m"oglichst
18 viele Umlaute enth"alt. Nat"urlich muss auch ein Zitat
19 her "'Kreative Menschen brauchen sinnfreie R"aume"'
20 (Quelle: unbekannt).
21
22 \section{Weiteres Kapitel}
23
24 Weiteren Text zu schreiben f"allt schwer,
25 aber es m"ussen mindestens zwei Abs"atze folgen,
26 die nat"urlich nicht zu kurz sein d"urfen.
27
28 \selectlanguage{english}
29 Proof if every letter and every digit is
30 included in the sentence from the \TeX{}
31 book ``the quick brown fox jumped over
32 9.876.543.210 lazy dogs''.

```

```

33 \selectlanguage{ngerman}
34 Wenn ein Dokument mehrere Kapitel hat, sollte man diese
35 in verschiedenen Dateien speichern und mit dem Befehl
36 \texttt{\textbackslash{}include}
37 einbinden. Hilfreich ist in diesem Zusammenhang
38 ebenfalls der Befehl \verb+\includeonly+.
39 Mehr dazu erfahren Sie im zweiten Band.
40
41 \selectlanguage{french}
42 \og{}Bien entendu, dans ce cas, le m\’ecanisme
43 multilingue utilis\’e est celui de Babel.
44 D’autres dispositifs globaux, impos\’es
45 par Babel, peuvent intervenir par ailleurs.\fg{}
46 (Bernard Gaulle (1998))
47
48 \selectlanguage{ngerman}
49 Zum Schluss noch ein Absatz wieder beginnend mit
50 \textbackslash{}selectlanguage\{ngerman\}
51 damit man Umlaute "a, "o, "u und "s verwenden
52 kann.
53

```

Inhaltsverzeichnis

1	Erstes Kapitel	1
2	Weiteres Kapitel	1

1 Erstes Kapitel

Und nun ein wenig seriöser Text, der möglichst viele Umlaute enthält. Natürlich muss auch ein Zitat her „Kreative Menschen brauchen sinnfreie Räume“ (Quelle: unbekannt).

2 Weiteres Kapitel

Weiteren Text zu schreiben fällt schwer, aber es müssen mindestens zwei Absätze folgen, die natürlich nicht zu kurz sein dürfen.

Proof if every letter and every digit is included in the sentence from the T_EX book “the quick brown fox jumped over 9.876.543.210 lazy dogs”.

Wenn ein Dokument mehrere Kapitel hat, sollte man diese in verschiedenen Dateien speichern und mit dem Befehl `\include` einbinden. Hilfreich ist in diesem Zusammenhang ebenfalls der Befehl `\includeonly`. Mehr dazu erfahren Sie im zweiten Band.

« Bien entendu, dans ce cas, le mécanisme multilingue utilisé est celui de Babel. D'autres dispositifs globaux, imposés par Babel, peuvent intervenir par ailleurs. » (Bernard Gaulle (1998))

Zum Schluss noch ein Absatz wieder beginnend mit `\selectlanguage{ngerman}` damit man Umlaute ä, ö, ü und ß verwenden kann.

Hilfreiche Ergänzungen

Unter dem Namen ‚Tools‘ sind eine Reihe von hilfreichen Paketen zusammengefasst, die bei keiner $\text{\LaTeX}$ -Installation fehlen sollten.

Pakete: afterpage, array, bm, calc, dcolumn, delarray, enumerate, fontsmpl, ftnright, hhline, indentfirst, layout, longtable, multicol, rawfonts, showkeys, somedefs, tabularx, theorem, trace, varioref, verbatim, xr, xspace

Darüber hinaus gehören zur Sammlung ‚Tools‘ einige $\text{\LaTeX}$ -Eingabedateien, die keinem der beschriebenen Pakete angehören. Die Dateien `e.tex`, `h.tex`, `q.tex`, `r.tex`, `s.tex` und `x.tex` haben die Aufgabe, bei einer fehlenden $\text{\LaTeX}$ -Eingabedatei die gleichen Angaben machen zu können wie bei anderen $\text{\LaTeX}$ -Fehlern. `e` steht für *edit*, `h` für *help*, `q` für *run quietly*, `r` für *run without stopping*, `s` für *scroll future errors* und `x` für *terminate*. Bei einer fehlenden Eingabedatei erwartet $\text{\LaTeX}$ nämlich den Namen der Datei, die an dieser Stelle einzulesen ist.

9.1 Paket afterpage

Dieses kleine Paket von David Carlisle ermöglicht es, Befehle nach Ausgabe der aktuellen Seite auszuführen, bevor mit der nächsten Seite begonnen wird. In einigen Fällen lassen sich damit Probleme bei der Formatierung verringern oder sogar vermeiden.

Zur Festlegung der nach der aktuellen Seite auszuführenden Befehle stellt das Paket den Befehl

```
\afterpage{<Befehl (e)>}
```

zur Verfügung. Im Argument werden die Befehle in der Reihenfolge angegeben, in der sie nach dem Ende der aktuellen Seite ausgeführt werden sollen.

Beispielsweise treten immer wieder Probleme mit größeren Tabellen oder Abbildungen auf, die nicht sofort gesetzt werden können und dann – zusammen mit allen folgenden Tabellen und Abbildungen – verschoben werden müssen. In diesen Fällen muss $\text{\LaTeX}$ sich den kompletten Inhalt merken, und das kann dann zu einem Fehler wegen mangelndem Speicherplatz führen. Das Auftreten eines auf diese Weise verursachten Fehlers lässt sich mit Hilfe eines `\clearpage`-Befehls vermeiden, der bekanntlich die Ausgabe aller noch bereits bearbeiteten, aber noch nicht ausgegebenen Gleitobjekte, also Tabellen und Abbildungen bewirkt. Allerdings wird durch diesen Befehl die aktuelle Seite auch – in der Regel vorzeitig – beendet.

```
\afterpage{\clearpage}
```

erlaubt es nun, den Befehl `\clearpage` zwischen das Ende der aktuellen und den Beginn der neuen Seite einzufügen, so dass die aktuelle Seite so beendet wird, wie es ohne `\clearpage` der Fall wäre. Vor Beginn der nächsten Seite werden alle anstehenden Gleitobjekte ausgegeben.

9.2 Paket array

Intern behandelt $\text{\LaTeX}$ die beiden Umgebungen `tabular` und `array` völlig gleich. Der einzige Unterschied besteht darin, dass `tabular` im Textmodus und `array` im mathematischen Modus verwendet wird. Die Inhalte der Spalten werden jeweils in dem Modus gesetzt, für den die Umgebung verwendet wird.

Obwohl dieses Paket von Frank Mittelbach und David Carlisle den Namen `array` trägt, erweitert es daher die beiden Umgebungen `array` und `tabular`. Insbesondere bietet es weitere Varianten von Spalten mit Zeilenumbruch sowie die Möglichkeit zur Definition eigener Spaltenkennungen zur Verwendung in der Definition einer Tabelle. Eine andere Erweiterung ist, dass am Beginn und Ende einer Spalte Befehle eingefügt werden können. Außerdem erlaubt das Paket die Verwendung zerbrechlicher Befehle in den Spalten, ohne dass diese mit `\protect` geschützt werden müssen. Hinzu kommen einige weitere Ergänzungen.

Ein – in der Regel kaum wahrnehmbarer – Unterschied der gesetzten Tabelle mit und ohne das Paket `array` besteht in der Behandlung der Linien. In der $\text{\LaTeX}$ -Definition der Tabelle beanspruchen Linien unabhängig

von ihrer Strichstärke keinen Platz. Damit ändert sich das Layout einer Tabelle durch das Einfügen *einfacher* Linien nicht. Dieser Vorteil wird bei dicken Linien (ab einer Strichstärke nahe des Spaltenabstands) allerdings zum Nachteil, da die Abstände zwischen der Linie und den Spalteneinträgen abhängig von der Strichstärke abnehmen und sogar die Linie den Eintrag überlagern kann. Das Paket array verfolgt eine andere Strategie und berücksichtigt die Strichstärke. Dadurch wird der Abstand zwischen einer Linie und dem Eintrag unabhängig von der Strichstärke. Bei dieser Strategie wird allerdings durch das Einfügen von Linien die Tabelle um die Strichstärken der eingefügten Linien größer.

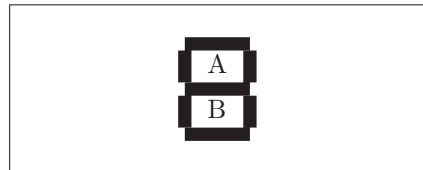
Diese unterschiedliche Strategie führt auch zu unterschiedlichen Abständen bei Mehrfachlinien. Bei normaler Strichstärke der Linien ist optisch kaum eine Abweichung festzustellen. Sehr deutlich wird der Unterschied aber, wenn die Strichstärke `\arrayrulewidth` verändert wird und in der Nähe des Abstands der Linien oder sogar darüber liegt. In diesem Fall verschwindet ohne array der Abstand zwischen den Linien, und die Linien fallen zu einer einzigen zusammen, da die Strichstärke nicht berücksichtigt wird. Mit array bleibt der Abstand wegen der berücksichtigten Strichstärke hingegen erhalten.

Ein weiterer Unterschied zeigt sich bei sehr dicken Linien. Im $\text{\LaTeX}$ -Standard erstrecken sich die horizontalen Linien bis in die Mitte der vertikalen Linien, da die Strichstärke nicht berücksichtigt wird. Die vertikalen Linien erstrecken sich nur zwischen den horizontalen Linien. Dadurch entstehen unschöne Abschlüsse, wenn eine horizontale Linie nur von einer Seite auf eine vertikale Linie trifft.

```

7  \setlength{\arrayrulewidth}%
8    {5pt}
9  \addtolength{\tabcolsep}%
10   {.5\arrayrulewidth}
11  \begin{tabular}{|l|}
12    \hline A \\
13    \hline B \\
14    \hline
15  \end{tabular}

```



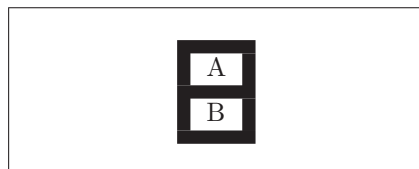
[Buch1/Ergaenz/arraybd-std.tex]

Mit dem Paket array tritt dieser Effekt wegen der Berücksichtigung der Strichstärke nicht auf, und die Ecken und Kanten werden korrekt gesetzt:

```

6 \usepackage{array}
7 \begin{document}
8 \setlength{\arrayrulewidth}{%
9   {5pt}
10 \begin{tabular}{|l|}
11 \hline A \\
12 \hline B \\
13 \hline
14 \end{tabular}
15 \end{document}

```



[Buch1/Ergaenz/arraybd-pkg.tex]

Wie man sieht ist die Eingabe der Tabellen völlig gleich. Die Strichstärke `\arrayrulewidth` beträgt 5 pt. Bei der Standard-**L**^AT_EX-Tabelle ist der Spaltenabstand `\tabcolsep` um `.5\arrayrulewidth` erhöht, um gleiche Abstände zu den Linien zu haben wie bei der mit `array` gesetzten Tabelle.

Häufig ist der Grund für die Verwendung des Pakets `array` der Wunsch oder die Notwendigkeit, seine erweiterten Möglichkeiten bei der Definition einer Tabelle zu nutzen. Die folgende Aufstellung führt der Vollständigkeit halber auch die unveränderten Angaben im Rahmen der Tabeledefinition auf.

Unverändert

- l** Linksbündig ausgerichtete Spalte,
- c** Zentriert ausgerichtete Spalte,
- r** Rechtsbündig ausgerichtete Spalte,
- p{<spbr>}** Spalte mit Zeilenumbruch der Breite <spbr>, oberste Zeile auf der Grundlinie (entspricht `\parbox[t]{<spbr>}`...),
- @{<spez>}** Trennung zweier Spalten durch <spez> statt durch den sonst verwendeten Spaltenzwischenraum.

Abweichendes Verhalten

- |** Trennung zweier Spalten durch eine senkrechte Linie (wie auch ohne das Paket). Im Unterschied zum Standard wird jedoch der Abstand zwischen den Spalten um die Linienstärke vergrößert.

Erweiterungen

- m{<spbr>}** Spalte mit Zeilenumbruch der Breite <spbr>, die vertikal gegenüber der Grundlinie zentriert ist (dies entspricht `\parbox{<spbr>}`...).
- b{<spbr>}** Spalte mit Zeilenumbruch der Breite <spbr>, unterste Zeile auf der Grundlinie (entspricht `\parbox[b]{<spbr>}`...).

- >{<spez>}** Kann vor einem Spaltentyp `l`, `c`, `r`, `p`, `m` oder `b` angegeben werden und fügt `<spez>` direkt vor dem Setzen der Spalte ein, so als ob der Inhalt der Spalte mit `<spez>` beginnen würde.
- <{<spez>}** Kann nach einem Spaltentyp `l`, `c`, `r`, `p`, `m` oder `b` angegeben werden und fügt `<spez>` direkt nach dem Setzen der Spalte ein, so als ob der Inhalt der Spalte mit `<spez>` enden würde.
- !{<spez>}** Wird wie `|` oder `@{<spez>}` verwendet und fügt `<spez>` statt `|` bei der Spaltentrennung ein. Im Unterschied zu `@{<spez>}` bleibt dabei der bei `|` verwendete zusätzliche Abstand erhalten.

Bei den Spaltentypen `p`, `m` und `b` mit Zeilenumbruch ist die Absatzeinrückung `\parindent` auf 0 mm eingestellt und damit unterdrückt. Sollen Absätze eingerückt werden, kann beispielsweise durch Angabe von

```
>{\setlength{\parindent}{5mm}}p{...}
```

eine Einrückung von 5 mm am Anfang von Absätzen eingestellt werden. Das folgende Beispiel zeigt diese unterschiedlich vertikal ausgerichteten Spalten mit Zeilenumbruch.

```

6  \usepackage{array}
7  \begin{document}
8  \begin{tabular}{>{\bfseries}c|c!{\vline}c@{\vline}c
9                p{1.6cm}m{1.6cm}b{1.6cm}>{\$}c<{\$}}
10 \multicolumn{8}{>{\Large\bfseries}l}{Spalte}\\
11 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8\\
12 \hline
13 \hline
14 12 & 34 & 56 & 78 & Text & Text & Text & \frac{a}{b}\\
15 \hline
16 abc & def & ghi & jkl & Spalte mit Zeilenumbruch
17   & Spalte mit Zeilenumbruch & Spalte mit Zeilenumbruch
18   & \sqrt{x} & \\
19 \hline
20 \end{tabular}
21 \end{document}
```

Spalte							
1	2	3	4	5	6	7	8
12	34	56	78	Text	Text	Text	$\frac{a}{b}$
abc	def	ghijkl	Spalte mit Zeilenum- bruch		Spalte mit Zeilenum- bruch	Spalte mit Zeilenum- bruch	$\sqrt{x}$

[Buch1/Ergaenz/arrayc.tex]

Die Angaben `>` und `<` waren anfänglich vor allem dazu gedacht, um in `tabular`- und `array`-Umgebungen wie bei der letzten Spalte im Beispiel zwischen dem Textmodus und dem mathematischen Modus wechseln zu können. Durch `>\$>c<\$>` wird eine einzelne Spalte im von der Umgebung der Tabelle abweichenden Modus gesetzt. Sie haben sich allerdings auch zur Umschaltung der Schriftart in einer Spalte – wie im Beispiel bei der ersten Spalte mit `>\bfseries>c` – bewährt. Selbst komplexe Angaben sind möglich, wobei es sich allerdings empfiehlt, einen Befehl zu definieren und als Argument anzugeben, der die gewünschten Einstellungen vornimmt.

Auch eine mehrfache Angabe von `>` und `<` für eine Spalte ist möglich. Die Angaben bei `>` werden allerdings in der umgekehrten Reihenfolge wirksam, d. h. `><Spez1>><Spez2>` entspricht `><Spez2><Spez1>`. Anlass für die Umkehr der Reihenfolge ist der Wunsch, auch zu einer mit `\newcolumntype` vordefinierten Spalte zusätzliche Angaben machen zu können. In diesem Fall wird der Anwender aber erwarten, dass die Angaben vor der Spalte eines selbstdefinierten Spaltentyps wirksam wird.

Der Unterschied zwischen den Angaben `@` und `!` wird besonders deutlich, wenn man die Wirkung der beiden Angaben `c!\hspace{5mm}>c` und `c@{\hspace{5mm}}>c` vergleicht. `!` vergrößert den Abstand der beiden Spalten um 5 mm, während `@` den Abstand der Spalten auf exakt 5 mm einstellt. Im Zusammenhang mit der Vorgabe der Tabellenbreite ist zu beachten, dass eine angegebene Spaltentrennung `@{...}` oder `!{...}` höchstens einen Befehl `\extracolsep` enthalten darf. Zusätzlich zu beachten ist, dass `\extracolsep` *explizit* angegeben sein muss. Die Angabe eines Makros, das diesen Befehl enthält, ist nicht möglich. Unter diesen Bedingungen kann `\extracolsep` auch bei der Definition eigener Spalten verwendet werden.

Im obigen Beispiel wird dieses Verhalten anhand der drei senkrechten Linien zwischen den ersten vier Spalten gezeigt. Die Spaltentrennung

zwischen den beiden ersten Spalten wird in der gewohnten Weise mit `|` vorgenommen. Die zweite und dritte Spalte werden durch `!\vline` getrennt; das Ergebnis unterscheidet sich nicht von der vorhergehenden Spaltentrennung und zeigt so, dass die üblichen Abstände erhalten bleiben. Bei der Spaltentrennung `@\vline` zwischen der dritten und vierten Spalte entfallen sie hingegen.

Neben den Erweiterungen der Angaben bei der Tabellendefinition stellt das Paket auch zusätzliche Befehle zur Verfügung.

Wird eine komplexe Spaltendefinition wie `>{<specs>}c{<spece>}` nicht nur einmalig, sondern in mehreren Tabellen verwendet, kann mit dem Befehl

```
\newcolumntype{<Name>}[<ArgNum>]{<Spezifikation>}
```

eine Spaltenbezeichnung `<Name>` vereinbart werden, die in gleicher Weise wie die vordefinierten Spaltentypen verwendet werden kann und so wirkt, als enthielte die Tabellendefinition statt `<Name>` die vollständige Spaltendefinition `<Spezifikation>`. Damit kann man sich die Wiederholung der Angaben in den einzelnen Tabellendefinitionen ersparen. Zu beachten ist, dass bei Fehlern in der Tabellendefinition diese so wiedergegeben werden, wie sie von $\text{\LaTeX}$ ausgewertet werden. Spalten, deren Typ mit `\newcolumntype` definiert wurden, werden daher nicht wie in der $\text{\LaTeX}$ -Quelle angezeigt, sondern in der bereits expandierten Form.

Das optionale Argument `<ArgNum>` hat die gleiche Bedeutung wie beim `\newcommand`-Befehl: `<ArgNum>` ist die Anzahl der Argumente, die in `<Spezifikationen>` zur Verfügung stehen und durch Angabe von `#<Num>` verwendet werden können, wobei `<Num>` die Nummer des Arguments ist.

`<Spezifikation>` enthält die Angaben, die zum Setzen der Spalte verwendet werden sollen. Zulässig sind alle Angaben, die eingangs für die Tabellendefinition aufgelistet wurden. Sie können sogar mehrfach in `<Spezifikation>` auftreten. Damit sind ‚Spaltendefinitionen‘ möglich, die durch Angabe einer einzelnen Spaltenbezeichnung mehrere Spalten – also eine Spaltengruppe – definieren. Für häufig auftretende Standardtabellen kann also die gesamte Tabellendefinition nach Vereinbarung mit dem `\newcolumntype`-Befehl durch Angabe der definierten Spaltenbezeichnung gesetzt werden. Für Spalten in Tabellen mit vorgegebener Tabellenbreite kann `<Spezifikation>` – wie oben beschrieben – auch einen expliziten `\extracolsep`-Befehl enthalten. Damit sind beim Befehl `\newcolumntype` sehr komplexe Vereinbarungen möglich.

Als Beispiel gibt die Dokumentation zum Paket die Definition von Spalten an, die in Tabellen Spalten im mathematischen Modus und in `array`-Umgebungen Textspalten setzen:

```
\newcolumnntype{C}{>{$}c<{$}}
\newcolumnntype{L}{>{$}l<{$}}
\newcolumnntype{R}{>{$}r<{$}}
\newcolumnntype{X}{clr}
```

Die neuen Spaltentypen C, L und R können wie die Standardtypen c, l und r in `tabular`- und `array`-Umgebungen bei der Definition verwendet werden. Der Spaltentyp x steht für eine Spaltengruppe bestehend aus drei Spalten – einer zentrierten, einer linksbündigen und einer rechtsbündigen Spalte – und dient damit als Abkürzung für die einzelnen Spaltenangaben.

Mit `\newcolumntype` vereinbarte Spaltentypen können – unabhängig davon, ob sie vom Anwender oder in einem Paket definiert wurden – mit dem Befehl

```
\showcols
```

am Bildschirm und in die Log-Datei ausgegeben werden. Neben der Bezeichnung für den Spaltentyp wird auch die zugehörige Definition ausgegeben.

Normalerweise werden Tabelle gegenüber der Grundlinie zentriert gesetzt. Durch Angabe der Option `⋈` kann man erreichen, dass die oberste Zeile auf der Grundlinie steht, und durch Angabe der Option `⋉`, dass die unterste Tabellenzeile auf der Grundlinie gesetzt wird. Das folgende Beispiel zeigt, dass diese Ausrichtung nur dann klappt, wenn die Tabelle nicht mit einer horizontalen Linie beginnt oder endet.

```

7 AB\quad
8 \begin{tabular}[t]{cc} cd & ef\\ 12 & 34 \end{tabular}
9 \quad GH\quad
10 \begin{tabular}{cc} ij & kl\\ 12 & 34 \end{tabular}
11 \quad MN\quad
12 \begin{tabular}[t]{cc}
13 \hline
14 op & qr\\ 12 & 34
15 \end{tabular}
16 \quad ST

```

AB	cd	ef	GH	ij	kl	MN	op	qr	ST
	12	34		12	34		12	34	

Das analoge Problem tritt auf bei einer Tabelle, die mit `\hline` endet und deren letzte Zeile auf der Grundlinie stehen soll.

Das Paket array löst diese Probleme mit den beiden Varianten des `\hline`-Befehls:

```
\firsthline
\lasthline
```

Die Befehle können jederzeit statt eines normalen `\hline`-Befehls vor der ersten Zeile (`\firsthline`) oder nach der letzten Zeile (`\lasthline`) verwendet werden. Sie führen dazu, dass bei Angabe der Optionen `t` oder `b` die erste oder letzte Zeile einer Tabelle korrekt auf der Grundlinie steht. Das folgende Beispiel zeigt das vorhergehende Beispiel unter Verwendung von `\firsthline` statt `\hline`.

```
8 \quad AB\quad
9 \begin{tabular}[t]{cc} cd & ef\\ 12 & 34 \end{tabular}
10 \quad GH\quad
11 \begin{tabular}{cc} ij & kl\\ 12 & 34 \end{tabular}
12 \quad MN\quad
13 \begin{tabular}[t]{cc}
14 \firsthline
15 op & qr\\ 12 & 34
16 \end{tabular}
17 \quad ST
18 \end{document}
```

AB	cd	ef	GH	ij	kl	MN		ST
	12	34		12	34		op	qr
							12	34

[Buch1/Ergaenz/arrayp2.tex]

Die Befehle `\firsthline` und `\lasthline` verwenden die im Paket definierte Länge

```
\extratabsurround
```

um zusätzlich Platz für die horizontale Linie zu reservieren. Die Länge hat einen voreingestellten Wert von 2.0 pt. Durch Vergrößern des Werts kann zusätzlich Platz bei `\firsthline` vor und bei `\lasthline` nach der Linie freigehalten werden.

Die Tabelle des Beispiels auf S. 233 zeigt, dass die Tabellenzeilen in der Höhe sehr knapp bemessen sind. Häufig ist es wünschenswert, die Abstände zu vorhergehenden und folgenden horizontalen Linien zu vergrößern. Den Abstand nach unten kann mit Hilfe des optionalen Arguments beim `\\`-Befehl leicht verändert werden. Den Abstand nach oben müsste man hingegen durch Einfügen eines unsichtbaren Zeichens, das weiter als die anderen Zeichen nach oben reicht, in jeder Zeile künstlich vergrößern. Das Paket `array` stellt nun eine zusätzliche Länge zur Verfügung, die den Abstand nach oben vergrößert, wenn sie positiv ist:

```
\extrarowheight
```

Der dieser Länge zugewiesene Wert führt über jeder Tabellenzeile zu einem zusätzlichen Abstand von `\extrarowheight`. Das folgende Beispiel zeigt die Wirkung dieses Befehls. Der Abstand wurde allerdings sehr groß gewählt, um die Wirkung deutlich zu zeigen.

```
8 \setlength{\extrarowheight}{5pt}
9 \begin{tabular}{>{\bfseries}c|c!{\vline}c@{\vline}c
10 p{1.6cm}m{1.6cm}b{1.6cm}>{\$}c<{\$}}
11 \multicolumn{8}{>{\Large\bfseries}l}{Spalte}\\
12 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8\\
13 \hline
14 \hline
15 12 & 34 & 56 & 78 & Text & Text & Text & \frac{a}{b}\\
16 \hline
17 abc & def & ghi & jkl & Spalte mit Zeilenumbruch
18 & Spalte mit Zeilenumbruch & Spalte mit Zeilenumbruch
19 & \sqrt{x} \\
20 \hline
```

Spalte							
1	2	3	4	5	6	7	8
12	34	56	78	Text	Text	Text	$\frac{a}{b}$
abc	def	ghi	jkl	Spalte mit Zeilenum- bruch	Spalte mit Zeilenum- bruch	Spalte mit Zeilenum- bruch	$\sqrt{x}$

[Buch1/Ergaenz/arrayh.tex]

Die Ausgabe zeigt, dass die in Spalte 6 mit `m{1.6cm}` gesetzte Spalte dabei gegenüber der Grundlinie im Vergleich zum Beispiel auf S. 233 ver-

schoben ist: Die mittlere Zeile steht nicht auf, sondern sichtbar unterhalb der Grundlinie. Durch Ändern der Spaltendefinition in

```
m{1.6cm}<{ \rule[-8.5pt]{0pt}{0pt}}
```

lässt sich der Effekt im Beispiel oben für die gesamte Tabelle korrigieren. Das Zeichen ~ wird dabei benötigt, um die Trennung des Worts „Zeilenumbruch“ zu ermöglichen. Die Korrekturlänge -8.5pt wurde durch Probieren gefunden.

```
8 \setlength{\extrarowheight}{5pt}
9 \begin{tabular}{>{\bfseries}c|c!{\vline}c@{\vline}c
10                p{1.6cm}m{1.6cm}<{\sim\rule[-8.5pt]{0pt}{0pt}}
11                b{1.6cm}>{\$}c<{\$}}
12 \multicolumn{8}{>{\Large\bfseries}l}{Spalte}\\
13 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8\\
14 \hline
15 \hline
16 12 & 34 & 56 & 78 & Text & Text & Text & \frac{a}{b}\\
17 \hline
18 abc & def & ghi & jkl & Spalte mit Zeilenumbruch
19 & Spalte mit Zeilenumbruch & Spalte mit Zeilenumbruch
20 & \sqrt{x} \\
21 \hline
```

Spalte							
1	2	3	4	5	6	7	8
12	34	56	78	Text	Text	Text	$\frac{a}{b}$
abc	def	ghi	jkl	Spalte mit Zeilenum- bruch	Spalte mit Zeilenum- bruch	Spalte mit Zeilenum- bruch	$\sqrt{x}$

[Buch1/Ergaenz/array2.tex]

Statt in der Spaltendefinition ist diese Korrektur natürlich auch in der Spalte selbst möglich. Ohne diese oder eine ähnliche Korrektur sollte man den Spaltentyp m vermeiden, wenn sich der Effekt störend auswirkt.

Die Erweiterungen der tabular-Umgebung ermöglichen Spalten mit Zeilenumbruch, bei denen die Zeilen zentriert oder mit Flatterrand statt im Blocksatz gesetzt werden. Dabei tritt jedoch ein Problem auf, weil

die Befehle `\centering`, `\raggedright` und `\raggedleft` den Befehl `\redefinieren`. Eine derartige Spalte wird beispielsweise in der Form `>\centering>p{<Breite>}` festgelegt. In dieser Spalte können die einzelnen Zeilen explizit mit `\` umbrochen werden. Dieses Verhalten wird durch den Übergang zur folgenden Spalte durch `&` beendet. Handelt es sich allerdings um die letzte Spalte in der Zeile muss der Abschluss der Tabellenzeile durch `\` erfolgen. Zur Lösung dieses Problems stellt array den Befehl

```
\tabularnewline
```

zur Verfügung, der in der beschriebenen Situation die Tabellenzeile abschließt.

9.3 Paket bm

Dieses Paket von David Carlisle mit Unterstützung von Frank Mittelbach stellt Befehle zum Setzen fetter mathematischer Symbole und Teilformeln zur Verfügung. Sind fette mathematische Zeichensätze geladen, verwendet bm diese Fonts. Das Paket sollte also erst *nach allen Fontpaketen* geladen werden. Wird ein fett zu setzendes Symbol in den vorhandenen Fonts nicht gefunden, verwendet bm ‚poor man’s bold‘ und überdruckt das Symbol mehrfach leicht versetzt.

Unterstützt werden nicht nur mathematische T_EX-Fonts und die Fonts der $\mathcal{A}\mathcal{M}\mathcal{S}$, sondern auch andere mathematische Schriftarten wie ‚math-time‘ und ‚Lucida New Math‘. Sogar erweiterte Fontsätze wie ‚mathtime plus‘ und ‚Lucida Expert‘ mit zusätzlichen *ultrafetten* Zeichen werden unterstützt. Begrenzt werden die Möglichkeiten zur Nutzung mathematischer Fonts vor allem dadurch, dass T_EX nicht mehr als 16 Fonts für Symbole und Buchstaben gleichzeitig zum Gebrauch im mathematischen Modus zulässt.

Das Laden des Pakets erfolgt mit den üblichen Befehlen. Optionen sind bei diesem Paket nicht vorgesehen.

```
\usepackage{bm}
```

Um im mathematischen Modus Teilformeln fett zu setzen, sind zwei Befehle vorgesehen:

```
\bm{<Teilformel>}
\hm{<Teilformel>}
```

Der erste Befehl `\bm` wählt die ‚normale‘ Fettschrift. `\hm` wählt die *ultrafette* Schrift, falls sie verfügbar ist; andernfalls verhält sich der Befehl wie `\bm`. Ist kein fetter mathematischer Font geladen, wird er durch leicht versetztes Überdrucken der Zeichen emuliert.

Bei den Klammersymbolen und anderen ‚großen‘ Symbolen ist zu beachten, dass häufig keine Fonts mit fetten Versionen von ihnen verfügbar sind. Das trifft nicht nur auf die ‚Computer Modern‘-Fonts zu, sondern auch auf die meisten PostScript-Fonts und ihre mathematischen Begleit-fonts wie ‚mathtime‘ oder ‚mathptm‘. Nur wenige Fontsätze wie ‚math-time plus‘ und ‚Lucida Expert‘ stellen solche großen Symbole fett oder sogar ultrafett zur Verfügung.

Die folgenden Beispiele zeigen einige Anwendungen des `\bm`-Befehls. Das Paket wird in Zeile 6 geladen und zeigt in den Zeilen 9 bis 13 das Verhalten des Befehls in verschiedenen Kombinationen.

```

6  \usepackage{bm}
7  \begin{document}
8  \[ 1 g \bm{g} \]
9  \[ 2 \mathrm{g}\bm{g} \]
10 \[ 3 {g}\bm{{g}} \]
11 \[ 4 \mathrm{{g}}\bm{{g}} \]
12 \[ 5 \mathrm{{g}}
13     \bm{\mathrm{g}} \]
14 \end{document}

```

*1gg**2gg**3gg**4gg**5gg*

[Buch1/Ergaenz/bmxml1.tex]

- In Zeile 8 wird `\bm` verwendet, um das zweite `g` nach der Ziffer 1 fett zu setzen.
- Zeile 9 zeigt nach der Ziffer 2 die gleiche Eingabe für die beiden Buchstaben `g` als Argument eines `\mathrm`-Befehls. In diesem Zusammenhang hat der `\bm`-Befehl offensichtlich keine Wirkung. Die Ursache dafür ist, dass der Befehl `\mathrm` auch auf die Teilformel `\bm{g}` wirkt und damit auch erzwingt, dass das Argument von `\bm` in der Schriftart `\mathrm` und nicht fett gesetzt wird.
- Zeile 10 entspricht Zeile 8, nur ist der Buchstabe `g` jeweils in geschweifte Klammern eingeschlossen. Die Ausgabe zeigt, dass die Ergebnisse übereinstimmen und die Klammern hier keinen Effekt haben.
- Schließt man in Zeile 11 analog zur 9ten Zeile die beiden Buchstaben in geschweifte Klammern ein, weicht das Ergebnis ab: Das zweite `g` ist in der fetten mathematischen Standardschrift ‚bold math italic‘ und nicht in `\mathrm` gesetzt. Das hat seinen Grund darin, dass `\bm`

sich bei komplexen Formeln – und dazu zählen in geschweifte Klammern eingeschlossene Teilformeln – nur die oberste Ebene hinsichtlich der Schriftart ausgewertet. Alles innerhalb geschweiften Klammern wird von `\bm` ohne Berücksichtigung abweichender Einstellungen wie hier durch den `\mathrm`-Befehl in der fetten mathematischen Schriftart gesetzt. `\mathrm{\bm{g}}` ist im Wesentlichen gleichbedeutend mit `\mathrm{\mbox{\boldmath$g$}}`, was die Ausgabe erklärt.

- Die Formel in den Zeilen 13, 14 zeigt jedoch, dass man die *Teilformel* `\mathrm{g}` mit `\bm` fett setzen kann. Analog zum vorhergehenden Fall entspricht der Befehl `\bm{\mathrm{g}}` einer Eingabe `\mbox{\boldmath$\mathrm{g}$}`.

```

8  \[
9  \sqrt{xyz} \quad
10 \bm{\sqrt{xyz}}
11 \]\[
12 \bm{\sqrt{\mbox{
13   \unboldmath$xyz$}}}
14 \quad
15 \bm{\sqrtsign{xyz}}
16 \]
```

$$\begin{array}{cc} \sqrt{xyz} & \bm{\sqrt{xyz}} \\ \bm{\sqrt{xyz}} & \bm{\sqrt{xyz}} \end{array}$$

[Buch1/Ergaenz/bmxmlpl2.tex]

Die Formeln zeigen verschiedene Kombinationen von `\bm` mit dem Wurzelzeichen. Die erste Wurzel ist normal, die zweite zusammen mit ihrem Argument mit Hilfe von `\bm` fett gesetzt. Es gibt allerdings keine einfache Möglichkeit, *nur das Wurzelsymbol* fett zu setzen, da es sich um einen Befehl mit einem Argument handelt und ein Befehl nicht allein, also ohne seine Argumente, Argument des `\bm`-Befehls sein kann. Eine Lösung für dieses Problem zeigen die Zeilen 12 und 13, in denen als Argument des `\sqrt`-Befehls eine `\mbox` verwendet wird, deren Inhalt im Textmodus gesetzt wird. In der `\mbox` kann dann eine Inline-Formel gesetzt werden, nachdem zuvor mit `\mathversion{normal}` oder `\unboldmath` (siehe 12.2.2) ausdrücklich eine nicht fette Darstellung der Formel ausgewählt wurde. Zeile 15 zeigt eine andere Lösung, bei der das in L^AT_EX vereinbarte Wurzelsymbol `\sqrtsign` mit `\bm` fett gesetzt wird im Unterschied zum folgenden Argument der Wurzel.

```

8  \[
9  \left[abc\right]
10 \quad
11 \bm{\left[{}abc\bm{\right]}}
12 \] \[
13 \left[\sqrt{abc}\right]
14 \quad
15 \bm{\left[{}\sqrt{abc}
16 \quad \bm{\right]}}
17 \]

```

[Buch1/Ergaenz/bmxml3.tex]

Das dritte Beispiel demonstriert das Verhalten von `\bm` im Zusammenhang mit Klammern. Zeile 9 enthält einen einfachen Ausdruck in eckigen Klammern. In Zeile 11 sind die Klammern mit `\bm` fett gesetzt. `\left`, `\right` oder andere Befehle, die die Größe der Klammern festlegen, müssen dabei innerhalb des Arguments von `\bm` stehen, weil ihnen eine Klammer folgen muss und `\bm{<Klammer>}` syntaktisch keine Klammer ist. Entscheidend dafür, dass die Klammern tatsächlich fett gesetzt werden, ist, dass die Klammern *als Zeichen* in einem fetten mathematischen Font verfügbar sind, und das ist nur bei *nicht* vergrößerten Klammern der Fall. Das zeigen die Zeilen 13 und 15/16, in denen die einzuklammernde Teilformel größere Klammern erzwingt. Vergrößerte Wurzelzeichen gehören übrigens wie die großen eckigen Klammern zu den ‚großen‘ Symbolen, die nicht fett verfügbar sind.

Der Befehl `\bm` wirkt ähnlich wie der `\boldsymbol`-Befehl aus dem Paket `amsbsy`, der auf S. 421 beschrieben ist. Um in einem Dokument ohne große Änderungen die beiden Pakete `amsbsy` und `bm` gegeneinander austauschen zu können, definiert das Paket `bm` die beiden Befehle

```

\boldsymbol{<Symbol>}
\heavysymbol{<Symbol>}

```

als Synonyme von `\bm{<Symbol>}` und `\hm{<Symbol>}`.

Der Satz ganzer Formeln in fetter Schrift ist mit dem in 12.2.2 (S. 421) beschriebenen Befehl `\boldmath` möglich. Das Paket `bm` definiert den Befehl

```

\heavymath

```

analog zu `\boldmath`, um Formeln *ultrafett* zu setzen. Ist kein ultrafetter mathematischer Font installiert, wird die Formel einfach fett gesetzt.

Das Paket `bm` treibt einigen Aufwand, um die Abstände im mathematischen Modus korrekt zu setzen, so dass `\bm{>}` als fettes Größerzeichen mit den von $\TeX$ für Relationen vorgesehenen Abständen wie ein einfaches Zeichen `>` gesetzt wird. Dieser Aufwand muss jedesmal betrieben werden, wenn ein `\bm-` oder `\hm-`Befehl verwendet wird. Um den Aufwand zu reduzieren, stellt das Paket Befehle zur Verfügung, mit denen mehrfach fett zu setzende Teilformeln einem Befehl zugeordnet und dann mehrfach ohne neuen Aufwand verwendet werden können. Der Grundbefehl zur Definition einer fetten Teilformel `<Formel>` hat die folgende Form

```
\DeclareBoldMathCommand[<Mversion>]{<Befehl>}{<Formel>}
```

und ist hauptsächlich zur Verwendung in Paketen gedacht. Er definiert den Befehl `<Befehl>`, mit dem die Teilformel fett gesetzt wird. Das optionale Argument `<Mversion>` legt fest, ob die Teilformel bei Angabe von `bold` fett oder im Fall von `heavy` ultrafett gesetzt wird. Fehlt das optionale Argument, wird `bold` angenommen.

Für den Anwender sind kürzere Formen des Befehls ohne Angabe des optionalen Arguments vorgesehen:

```
\bmdefine{<Befehl>}{<Formel>}
\hmdefine{<Befehl>}{<Formel>}
```

`\bmdefine` ist eine Abkürzung für `\DeclareBoldMathCommand[bold]` und `\hmdefine` für `\DeclareBoldMathCommand[heavy]`.

Es wurde eingangs bereits darauf hingewiesen, dass $\TeX$ den gleichzeitigen Gebrauch von Fonts im mathematischen Modus auf 16 beschränkt. Diese Beschränkung gilt für die mathematische Version `normal` ebenso wie für `bold`. Um möglichst einfach und schnell auf die fetten und ultrafetten Fonts zugreifen zu können, installiert `bm` sie in der mathematischen Version `normal`. Wegen der Beschränkung werden *maximal vier* fette Fonts und *maximal drei* ultrafette Fonts installiert. Vier weitere mathematische Fonts werden generell von $\LaTeX$ geladen. Bei Verwendung eines erweiterten mathematischen Fontsatzes wie ‚`mathtime plus`‘, der auch ultrafette Fonts verwendet, wird zusätzlich ein Symbolfont installiert, so dass insgesamt bereits 12 Fonts verbraucht sind. Für weitere mathematische Fonts wie die Symbolfonts der $\mathcal{M}$ S und `\math...` verbleiben dann lediglich vier weitere Fonts.

Um mehr Platz für zusätzliche Fonts zu lassen, sind Befehle vorgesehen, die die Anzahl der von `bm` geladenen fetten und ultrafetten Fonts beschränken. Es handelt sich dabei um zwei Befehle, die beim Laden des

Pakets `bm` berücksichtigt werden, und die vor dem `\usepackage`-Befehl mit `\newcommand` zu definieren sind:

```
\bmmax
\hmmax
```

Der Voreinstellung würden die Vereinbarungen

```
\newcommand{\bmmax}{4}
\newcommand{\hmmax}{3}
```

entsprechen. Stehen ultrafette Fonts zur Verfügung, die man aber nicht verwenden will, kann man mit

```
\newcommand{\hmmax}{0}
```

die Installation der drei ultrafetten Fonts verhindern.

Setzt man `\bmmax` auf 0, so verhindert man die Installation der fetten Fonts in der mathematischen Version `\normal`. Der `\bm`-Befehl setzt sein Argument allerdings trotzdem im korrekten fetten Font, indem er die mathematische Version `\bold` verwendet, was etwas länger dauert und auch mit Klammern nicht funktioniert. Andererseits kann gegebenenfalls die Anzahl fetter oder ultrafetter Fonts hochgesetzt werden, um zusätzliche fette oder ultrafette Fonts zu installieren.

Der Befehl `\bm` muss das angegebene Argument analysieren, um die Abstände korrekt zu setzen. Enthält das Argument nur Symbole, die in $\text{\LaTeX}$ selbst und seinen Grundpaketen oder in den Ergänzungen der $\mathcal{A}\mathcal{M}\mathcal{S}$ definiert sind, sollten keine Probleme auftreten. Werden im Argument jedoch ungewöhnliche Konstruktionen verwendet, kann es passieren, dass die Analyse nicht korrekt durchgeführt werden kann. In der Folge treten höchstwahrscheinlich ungewöhnliche Fehler auf, die zu einem Abbruch führen. Zur Abhilfe empfiehlt die Dokumentation, die den Fehler verursachende(n) Teilformel(n) in geschweifte Klammern einzuschließen, da `\bm` bei in geschweifte Klammern eingeschlossenen Teilformeln nicht versucht, Abstände selbst korrekt zu setzen. Daher wird bei solchen Teilformeln auf eine Analyse verzichtet und der Fehler in der Regel vermieden. Verursacht beispielsweise ein Befehl `<\cmd>` den Fehler, lässt er sich in der Regel durch `\bm{\{<\cmd>\}}` umgehen. Für die korrekten Abstände muss der Anwender gegebenenfalls selbst sorgen, etwa durch `\bm{\mathrel{\cmd}}`, wenn die Abstände wie bei einer Relation gesetzt werden sollen.

9.4 Paket calc

Das Paket calc von Kresten Krab Thorup und Frank Jensen unter Mithilfe von Chris Rowley vereinfacht die Berechnung von Längen und Zählerwerten. Um das zu erreichen, werden die Befehle `\setcounter`, `\addtocounter`, `\setlength` und `\addtolength` so redefiniert, dass für die Werte einfache Ausdrücke angegeben werden können. Bei Zählern ist dies ein ganzzahliges Ergebnis und bei Längenbefehlen eine Länge. Neben den genannten Befehlen können einfache Längenausdrücke in den meisten Fällen auch dort angegeben werden, wo als Wert ein Längenmaß angegeben werden kann oder muss, beispielsweise bei `\makebox`- und `\parbox`-Befehlen oder bei Umgebungen wie `minipage` oder `p`-Spalten in `tabular`-Umgebungen, da diese Befehle und Umgebungen die Längen mit einem `\setlength`-Befehl übernehmen. Abgesehen von der Möglichkeit der Angabe eines Ausdrucks haben die Befehle und Umgebungen ihre gewohnte Form und müssen an dieser Stelle nicht neu vorgestellt werden.

„Einfache Ausdrücke“ im Sinne dieses Pakets setzen sich aus Zahlangaben und Längenmaßen, den vier Grundrechenarten `+`, `-`, `*` und `/` und gegebenenfalls runden Klammern zusammen. Die Grundrechenarten haben ihre gewohnten Prioritäten „Punkt vor Strich“; `*` und `/` binden also stärker als `+` und `-`. Ein Teilausdruck, der aus mehreren Operationen gleicher Priorität besteht, wird von links nach rechts ausgewertet. Mit Hilfe der Klammern kann eine abweichende Reihenfolge der Operationsausführung festgelegt werden.

Bei Längen muss der erste angegebene Operand eine Zahl mit einer Maßeinheit sein, etwa `4cm*4/3`. Die getestete Version von calc unterscheidet allerdings nicht zwischen Befehlen und Längen und erkennt daher gegebenenfalls nicht, dass eine Länge berechnet wird. Dieses Problem lässt sich durch Angabe eines ersten Summanden mit `0pt` umgehen. Nur so lässt sich sicherstellen, dass bei der Auswertung von Anfang an bekannt ist, ob das (temporäre) Ergebnis einer Länge oder einem Zähler zugewiesen werden muss. Darüberhinaus müssen bei Längensummen alle Summanden mit einer Maßeinheit versehen sein, beispielsweise `2cm+4pt`.

Multiplikation mit und Division durch ganze Zahlen wird wie üblich angegeben. Ist ein Operand eine reelle Zahl, muss eine spezielle Form verwendet werden:

```
\real{<Dezimale Konstante>}
```

Das Argument *<Dezimale Konstante>*, bestehend aus Ziffern und einem Dezimalpunkt, kann mit einem Vorzeichen versehen sein. Ebenfalls als reelle Zahl wird

```
\ratio{<Längenausdr a>}{Längenausdr b}
```

angesehen, der das Verhältnis $\text{<Längenausdr a>/<Längenausdr b>}$ der beiden Längenausdrücke beschreibt.

Zum Abmessen von Textpassagen stellt L^AT_EX die Befehle `\settowidth`, `\settoheight` und `\settddepth` zur Verfügung. Im Zusammenhang mit der formelmäßigen Behandlung von Längen wäre es allerdings störend, wenn die gemessenen Werte zuerst einer Länge zugewiesen und dann erst verwendet werden könnten. Daher stellt das Paket calc folgende Befehle zur Verfügung, die ohne Zuweisung an eine Länge in den von calc unterstützten Formeln verwendet werden können:

```
\widthof{<Text>}
\heightof{<Text>}
\depthof{<Text>}
```

Die Befehle setzen die Breite (`\widthof`), Höhe über der Grundlinie (`\heightof`) und die Unterlänge (`\depthof`) der als Argument angegebenen Textpassage ein. Statt

```
\settowidth{\textwd}{Einige Worte Text}
\setlength{\parindent}{.15\textwd}
```

kann man also kürzer

```
\setlength{\parindent{\widthof{Einige Worte Text} *
\real{0.15}}}
```

schreiben.

Bei Zählern werden berechnete Werte durch Abschneiden der Nachkommastellen auf ganze Zahlen gerundet. Es wird aber nicht nur das Endergebnis durch Abschneiden gerundet, sondern auch alle Zwischenergebnisse. Das Ergebnis einer Berechnung kann bei Zählern daher deutlich vom erwarteten Ergebnis abweichen. Auch die Reihenfolge der Operationen kann dabei das Ergebnis wesentlich beeinflussen, wie das folgende Beispiel zeigt:

```
3 · 1.6 · 1.7 = 8.16
3 * \real{1.6} * \real{1.7} = 4 * \real{1.7} = 6
3 * \real{1.7} * \real{1.6} = 5 * \real{1.6} = 8
3 * (\real{1.6} * \real{1.7}) = 3 * 2 = 6
```

9.5 Paket dcolumn

Bei Tabellen (oder Matrizen im mathematischen Modus) sieht $\text{\LaTeX}$ nur links- oder rechtsbündige und zentrierte Spalten vor. Eine Ausrichtung am Dezimalkomma (oder -punkt) ist nicht vorgesehen und muss gegebenenfalls mit zusätzlichem Aufwand simuliert werden.

Das Paket `dcolumn` von David Carlisle erweitert die `tabular`-Umgebung um einen Spaltentyp für Zahlen, die an einem vorgegebenen Trennzeichen wie dem Dezimalkomma ausgerichtet werden. Die neue Spaltendefinition hat die folgende Form:

```
D{<Trennzeichen-TeX>}{<Trennzeichen-dvi>}{<Format>}
```

`<Trennzeichen-TeX>` ist dabei ein einzelnes Trennzeichen in der $\text{\LaTeX}$ -Eingabedatei, an dem die Zahlen ausgerichtet werden. Es kann ein Buchstabe oder ein Sonderzeichen (ohne Sonderbedeutung für $\text{\LaTeX}$) verwendet werden. Leerzeichen oder Tabulatoren sind nicht erlaubt. Statt `<Trennzeichen-TeX>` wird im Dokument `<Trennzeichen-dvi>` gesetzt. Da beides vorgeschriebene Argumente sind, müssen sie auch dann angegeben werden, wenn sie gleich sind. `<Trennzeichen-dvi>` kann auch $\text{\LaTeX}$ -Code enthalten. Zu beachten ist, dass sowohl die Zahlen als auch das `<Trennzeichen-dvi>` im mathematischen Modus gesetzt werden.

`<Format>` legt fest, wieviel Platz für die Zahl in der zweiten Spalte reserviert wird. Die Angabe einer natürlichen (positiven) Zahl oder von 0 legt die Anzahl der Nachkommastellen fest. Die am Trennzeichen ausgerichteten Zahlen werden rechtsbündig unter Berücksichtigung der zulässigen Nachkommastellen gesetzt. Die Spaltenbreite wird so gewählt, dass die maximale Anzahl der Vorkommastellen in der Spalte gesetzt werden kann.

Eine negative Zahl lässt hingegen eine beliebige Anzahl von Nachkommastellen zu. In diesem Fall wird die Spalte so ausgerichtet, dass das Trennzeichen in der Spalte zentriert ist. Die Spaltenbreite wird so gewählt, dass die maximale Anzahl von Vor- und Nachkommastellen in der Spalte Platz findet. Unterscheidet sich die maximale Anzahl der Vorkommastellen von der der Nachkommastellen, bleibt auf einer Seite des Trennzeichens ungenutzter Platz frei.

Daher wird zusätzlich die Möglichkeit geboten, sowohl die Anzahl der Vorkommastellen als auch die der Nachkommastellen festzulegen. In diesem Fall hat die Formatangabe die Form

```
<Vorkommastellen><Trenner><Nachkommastellen>
```

mit (positiven) Zahlen $\langle\text{Vorkommastellen}\rangle$ und $\langle\text{Nachkommastellen}\rangle$. Als $\langle\text{Trenner}\rangle$ kann ein beliebiger Buchstabe oder ein Sonderzeichen (ohne Sonderbedeutung für $\text{\LaTeX}$) verwendet werden. Ziffern, Leerzeichen oder ein Tabulator können nicht verwendet werden. Es liegt nahe, $\langle\text{Trennzeichen-TeX}\rangle$ als $\langle\text{Trenner}\rangle$ anzugeben. Der gesamte Zahlenblock wird zentriert gesetzt. Werden mehr Vor- oder Nachkommastellen angegeben als tatsächlich genutzt, bleibt der für sie reservierte Platz frei. Damit kann man auch den Abstand zu einer links oder rechts stehenden Spalte vergrößern.

In diesem Beispiel wird in Zeile 6 das Paket dcolumn geladen. Die beiden Tabellen unterscheiden sich lediglich in der Definition der dezimal ausgerichteten Spalten und in einem Zelleintrag (3te Spalte erste Zeile). Dadurch soll die Ausrichtung der dezimalen Spalten gegenüber den Kopfzeilen deutlich gemacht werden.

```

6  \usepackage{dcolumn}
7  \begin{document}
8  \begin{center}
9  \begin{tabular}{|D{.}{,}{2}|D{,}{,}{3,1}|D{.}{\cdot}{-1}|}
10 \multicolumn{1}{|c|}{\textbf{Spalte 1}}
11 & \multicolumn{1}{|c|}{\textbf{Spalte 2}}
12 & \multicolumn{1}{|c|}{\textbf{Spalte 3}} \\
13 \hline
14 123.45 & 1,2 & 1.23 \\
15 1 & 100 & 1234.5 \\
16 .02 & 99,9 & 66.3
17 \end{tabular}
18 \end{center}
19
20 \begin{center}
21 \begin{tabular}{|D{.}{,}{5.3}|D{,}{,}{1}|D{.}{\cdot}{-1}|}
22 \multicolumn{1}{|c|}{\textbf{Spalte 1}}
23 & \multicolumn{1}{|r|}{\textbf{Spalte 2}}
24 & \multicolumn{1}{|c|}{\textbf{Spalte 3}} \\
25 \hline
26 123.45 & 1,2 & 1.23456 \\
27 1 & 100 & 1234.5 \\
28 .02 & 99,9 & 66.3
29 \end{tabular}
30 \end{center}
31 \end{document}

```

Spalte 1	Spalte 2	Spalte 3
123,45	1,2	1·23
1	100	1234·5
,02	99,9	66·3

Spalte 1	Spalte 2	Spalte 3
123,45	1,2	1·23456
1	100	1234·5
,02	99,9	66·3

[Buch1/Ergaenz/dcolumn.tex]

Die erste Spalte zeigt in der oberen Tabelle eine rechtsbündige mit zwei Nachkommastellen ausgerichtete Spalte. Bei der unteren Tabelle ist die Anzahl der Vor- und Nachkommastellen vorgegeben, um Platz für zusätzliche, aber nicht genutzte Stellen in dieser ebenfalls rechtsbündig gesetzten Spalte zu reservieren. Wegen der Länge der Spaltenüberschrift wirkt sich das allerdings nur bei den Nachkommastellen aus und führt so zu einer Verschiebung der Spalte um eine Ziffernbreite nach links. In beiden Tabellen wird der Dezimalpunkt in der Eingabe als Dezimalkomma gesetzt.

Bei der zweiten Spalte wird das Dezimalkomma in der Eingabe beibehalten. Die Formatangabe ist – bei gleicher Anzahl von Nachkommastellen – ähnlich wie in der ersten Spalte, aber vertauscht. Dass die Spalte in der oberen Tabelle zentriert und in der unteren rechtsbündig angeordnet ist, liegt an der unterschiedlichen Ausrichtung der Kopfzeile: Zentriert in der oberen (Zeile 11) und rechtsbündig in der unteren Tabelle (Zeile 23).

Die dritte Spalte ist in beiden Tabellen gleich. Der Dezimalpunkt der Eingabe wird als mathematisches Zeichen `\cdot` gesetzt. Wie eingangs erwähnt, werden die dezimalen Spalten und `<Trennzeichen-dvi>` im mathematischen Modus gesetzt. Der wesentliche Unterschied zu den beiden anderen Spalten ist jedoch die Formatangabe `-1`, die zu einer Spalte führt, die bezüglich des Trennzeichens zentriert ist, also gleich viele Vor- und Nachkommastellen vorsieht. Dies zeigt der unterschiedliche Eintrag in der ersten Datenzeile beider Tabellen: Die untere Tabelle enthält in der dritten Spalte drei zusätzliche Nachkommastellen, die die Spalte verbreitern und damit wegen der Symmetrie Platz für eine zusätzliche Vorkommastelle reservieren.

Werden häufiger Tabellen mit am Dezimalkomma ausgerichteten Spalten verwendet, kann der `\newcolumn`-Befehl verwendet werden, um in den einzelnen Tabellen diese Spalten einfacher und einheitlich zu vereinbaren. Das folgende Beispiel zeigt eine kleine Anwendung:

```

8 \newcolumntype{K}{D{,}{,}{2}}
9 \newcolumntype{L}[1]{D{,}{\cdot}{#1}}
10 \begin{tabular}{|L{-1}|L{3.3}|*{2}{K|}}
11 123,45678 & 1,23 & 3,21 & 12,34 \\
12 2,34 & 123,4 & 1 & ,12 \\
13 0,99 & 22,33 & 123,45 & 1234,56
14 \end{tabular}

```

	123.45678	1.23	3,21	12,34
	2.34	123.4	1	,12
	0.99	22.33	123,45	1234,56

[Buch1/Ergaenz/dcolumn2.tex]

Die senkrechten Linien dienen in diesem Beispiel nur zur Verdeutlichung der Spaltenbreiten, sie sind bei tabelliertem Material keineswegs zwingend erforderlich.

9.6 Paket delarray

Die `array`-Umgebung im mathematischen Modus wird häufig mit an die Höhe des `arrays` angepassten Klammern verwendet. David Carlisle hat das Paket `delarray` entwickelt, Kurzform für *delimited array*, um solche Umgebungen einfacher eingeben zu können. Die vereinfachte Eingabe eines `arrays` mit Begrenzern hat die folgende Form:

```

\begin{array}[LB]{<Spaltenvereinbarung>}[RB]
  <Tabellenzeilen> ...
\end{array}

```

Klammern können also direkt im Zusammenhang mit der `array`-Definition in der im mathematischen Modus üblichen Weise angegeben werden. Im Zusammenspiel mit dem Paket `array` (s. 9.2, S. 9.2) stehen auch die dort definierten Erweiterungen zur Verfügung.

Das folgende Beispiel zeigt links die vereinfachte (Zeilen 9–13) und rechts die normale Eingabe (Zeilen 15–19).

```

6 \usepackage{delarray}
7 \begin{document}
8 \[
9 \begin{array}{|ccc|}
10 a & b & c \\
11 b & a & d \\
12 c & d & a
13 \end{array}
14 \qquad
15 \left| \begin{array}{ccc}
16 a & b & c \\
17 b & a & d \\
18 c & d & a
19 \end{array} \right|
20 \]
21 \end{document}

```

$$\begin{array}{|ccc|} a & b & c \\ b & a & d \\ c & d & a \end{array} \qquad \left| \begin{array}{ccc} a & b & c \\ b & a & d \\ c & d & a \end{array} \right|$$

[Buch1/Ergaenz/delarray1.tex]

Auch in der Definition von `delarray` steht die ‚unsichtbare Klammer‘ als `.` (Punkt) zur Verfügung, wenn ein `array` nur auf einer Seite mit einem sichtbaren Begrenzer versehen werden soll.

Die vertikale Ausrichtung durch ein optionales Argument ist weiterhin möglich und führt nicht zu einem Konflikt mit einer öffnenden eckigen Klammer als linkem Begrenzer.

```

8 \[
9 \begin{array}{[t][c]}
10 x \\ y \\ z
11 \end{array}
12 \begin{array}{[c][c]}
13 x \\ y \\ z
14 \end{array}
15 \begin{array}{[b][c]}
16 x \\ y \\ z
17 \end{array}
18 \]

```

$$\begin{array}{[t][c]} x \\ y \\ z \end{array} \begin{array}{[c][c]} x \\ y \\ z \end{array} \begin{array}{[b][c]} x \\ y \\ z \end{array}$$

[Buch1/Ergaenz/delarray2.tex]

Nun wird zusätzlich gezeigt, dass Standard- $\text{\LaTeX}$, bei der Ausrichtung an der Grundlinie andere Ergebnisse liefert.

```

8  \[
9  \left[\begin{array}{t}{c}
10 x\\y\\z
11 \end{array}\right]
12 \left[\begin{array}{c}{c}
13 x\\y\\z
14 \end{array}\right]
15 \left[\begin{array}{b}{c}
16 x\\y\\z
17 \end{array}\right]
18 \]
```

[Buch1/Ergaenz/delarray3.tex]

Das Beispiel zeigt, dass die Klammern der `arrays` in den beiden Varianten unterschiedliche Größen haben. Während sie auf der linken Seite in der vereinfachten Form genau auf die Zeilen abgestimmt ist, erstrecken sich die Klammern bei den beiden äußeren `arrays` in der zweiten Ausgabezeile über die komplette Höhe der Zeile mit den insgesamt fünf Zelleinträgen. Wenn diese Ausgabe bei der vereinfachten Form im Paket `delarray` gewünscht würde, müssten leere `array`-Zeilen eingefügt werden.

9.7 Paket enumerate

Mit Hilfe dieses Pakets von David Carlisle lässt sich die Marke in einer `enumerate`-Liste in allgemeinerer Form festlegen. Dabei kann man nicht nur in sehr einfacher Weise die gewünschte Darstellung des Zählers auswählen, sondern die Kennung des Eintrags auch durch (kurze) Texte ergänzen.

Abgesehen vom Laden des Pakets sind im Dokument lediglich kleine Ergänzungen in den Listen erforderlich, in denen die Erweiterungen des Pakets genutzt werden. Zu diesem Zweck wird die `enumerate`-Umgebung um ein optionales Argument erweitert:

```

\begin{enumerate}[<Markenspez>]
  \item...
\end{enumerate}
```

Die `<Markenspez>` kann beliebigen Text enthalten, der auch $\LaTeX$ -Befehle enthalten darf. Die Position und die Darstellung des Zählers wird durch einen Buchstaben festgelegt, der nicht Teil einer Gruppe oder eines Arguments ist. Textteile, die einen der dafür vorgesehenen Buchstaben enthalten, müssen daher in geschweifte Klammern eingeschlossen sein.

Für die Darstellung des Zählers sind die folgenden Buchstaben vorgesehen:

A \Alph	I \Roman	1 \arabic
a \alph	i \roman	

Querverweise mit den Befehlen `\label` und `\ref` verhalten sich wie üblich. Anzumerken ist allerdings, dass der `\ref`-Befehl nur die Darstellung des Zählers liefert; die hinzugefügten Texte werden unterdrückt.

```

9  \begin{enumerate}[zu 1)]
10 \item Angaben zum ersten
11     Punkt\label{top:a}
12 \item vgl.~\ref{top:a})
13 \item Keine weiteren Angaben
14 \end{enumerate}

```

zu 1) Angaben zum ersten Punkt
zu 2) vgl. 1)
zu 3) Keine weiteren Angaben

[Buch1/Ergaenz/enumext1.tex]

Die Markenvereinbarung der Liste besteht scheinbar aus einem einfachen Text. Auf die Ziffer 1 trifft das allerdings nicht zu. Die Ziffer steht vielmehr für die Darstellung des Zählers als normale Zahl. Trotzdem kann man auf Gruppenklammern verzichten, da der übrige Text keinen der für die Darstellung des Zählers vorgesehenen Buchstaben enthält.

```

9  \begin{enumerate} [{ad} i:]
10 \item \texttt{A a I i} und
11     \texttt{1} stehen
12     f"ur verschiedene
13     Darstellungen des
14     Z"ahlers.
15 \item Daher muss \texttt{ad}
16     in geschweifte Klammern
17     gesetzt werden.
18 \end{enumerate}

```

ad i: A a I i und 1 stehen für verschiedene Darstellungen des Zählers.

ad ii: Daher muss ad in geschweifte Klammern gesetzt werden.

[Buch1/Ergaenz/enumext2.tex]

Bei diesem zweiten Beispiel werden die geschweiften Klammern im optionalen Argument benötigt, da a (für die Darstellung des Zählers durch Kleinbuchstaben) im Text vorkommt. Entsprechend der Angabe i wird der Zähler in dieser Liste durch römische Ziffern (mit Kleinbuchstaben) eingesetzt.

Wie man hier sehen kann, werden die Marken rechtsbündig in den mit `\labelwidth` festgelegten Raum platziert. Eine Änderung ist bei den Standardlisten und auch mit dem Paket `enumerate` nicht vorgesehen. Das Paket `expdlist` bietet eine Alternative.

9.8 Paket fontsmpl

Dieses Paket von Alan Jeffrey ermöglicht eine Beispielausgabe des aktuellen Fonts. Es dient damit ausschließlich der Beurteilung eines Fonts, den man in einem Dokument verwenden möchte. Es wird mit

```
\usepackage{fontsmpl}
```

geladen. Paketoptionen sind nicht vorgesehen.

Der Befehl zur Ausgabe lautet:

```
\fontsample
```

Er hat keine Argumente.

Nach Ausgabe des verwendeten Fonts wird ein englischer Beispieltext ausgegeben, der auch einige ‚Fremdwörter‘ enthält, um den Gebrauch von Akzenten und Ligaturen zu demonstrieren. Es folgt eine Reihe von Sonderzeichen wie `\copyright` oder `\texbullet`, die zusammen mit den Befehlen zu ihrer Ausgabe gezeigt werden. Den Abschluss bildet eine Liste von Akzenten, die jeweils auf eine Reihe von Groß- und Kleinbuchstaben angewendet wird.

```
1 \documentclass{article}
2 \usepackage[T1]{fontenc}
3 \setlength{\textwidth}{11cm}
4 \setlength{\textheight}{4cm}
5 \setlength{\parindent}{0cm}
6 \pagestyle{empty}
7 \usepackage{fontsmpl}
8 \usepackage{times}
9 \begin{document}
10 \fontsample
11 \end{document}
```

Test of font T1/ptm/m/n. Some text:

On November 14, 1885, Senator & Mrs. Leland Stanford called together at their San Francisco mansion the 24 prominent men who had been chosen as the first trustees of The Leland Stanford Junior University. They handed to the board the Founding Grant of the University, which they had executed three days before. This document—with various amendments, legislative acts, and court decrees—remains as the University’s charter. In bold, sweeping language it stipulates that the objectives of the

Das Beispiel zeigt nur einen kleinen Ausschnitt der tatsächlichen Ausgabe. Der gewählte Font ist durch das Paket `times` vorgegeben.

Neben der Paketversion für eigene Tests steht auch eine Datei `fontsmpl.tex` zur Verfügung. Dieses $\text{\LaTeX}$ -Dokument ist interaktiv und erwartet die Eingabe einer Fontfamilie. Die oben beschriebene Beispielausgabe wird jeweils für die Kodierungen `T1` und `OT1` für den Grundfont sowie die Fontvarianten `\itshape`, `\slshape`, `\scshape` in normaler Strichstärke und fett ausgegeben. Man erhält also eine Übersicht über die gesamte Fontfamilie in den beiden Standardkodierungen.

9.9 Paket `ftnright`

Der Inhalt dieses Pakets von Frank Mittelbach ist die Behandlung von Fußnoten im zweispaltigen Satz. Ohne das Paket werden Fußnoten zu einer Spalte unterhalb der Spalte gesetzt. Bei mehrspaltigem Satz mit Hilfe des Pakets `multicol` werden die Fußnoten unterhalb aller Spalten einspaltig gesetzt. In ersten Fall lässt der optische Eindruck zu wünschen übrig, wenn viele Fußnoten vorhanden sind. Die einspaltigen Fußnoten beim Paket `multicol` verschwenden Platz bei mehreren kurzen Fußnoten.

Aus diesen Überlegungen heraus wurde das Paket `ftnright` entwickelt, das die Fußnoten zu den beiden Spalten bei Angabe der Option `twocolumn` in der `\documentclass`-Anweisung an das Ende der rechten Spalte setzt. Ein derartiges Layout stand, obwohl bereits gelegentlich in Büchern umgesetzt, in $\text{\LaTeX}$ zunächst nicht zur Verfügung.

Beim Paket `multicol` kann die Anzahl der Spalten selbst auf einer einzelnen Seite variiert werden. Abgesehen von der Frage der Platzierung der Fußnoten treten dabei sehr große technische Probleme auf. Aus diesem Grund kann das Paket `ftnright` nicht zusammen mit `multicol` verwendet werden.

Der Block der Fußnoten in der letzten Spalte wird durch einen Abstand von der eigentlichen Spalte getrennt. Der sonst in $\text{\LaTeX}$ in der Regel verwendete Markierungsstrich hat standardmäßig die Breite 0 mm und fehlt daher. Gegebenenfalls kann er mit Hilfe des Befehls `\footnoterule` eingefügt werden, aber es sollte darauf geachtet werden, dass er keinen Platz beansprucht (etwa durch Hinzufügen der korrigierenden negativen Abstände).

Die Fußnoten werden in einer kleineren Schrift gesetzt. Die Fußnotenmarkierungen werden an dieser Stelle nicht hochgestellt gesetzt, sondern als Zahlen auf der Grundlinie in der für die Fußnoten verwendeten Schrift.

Das Paket ftnright verändert nur das Layout und stellt keine speziellen Befehle zur Verfügung. Es wird dringend empfohlen, dieses Paket als letztes zu laden. Abgesehen vom Laden des Pakets gibt es keine Änderungen im Dokument.

Zunächst zum Vergleich Fußnoten im zweispaltigen Standardlayout:

Der Block der Fußnoten in der rechten Spalte wird durch einen Abstand von der eigentlichen Spalte¹ getrennt.

Der sonst in L^AT_EX in der Regel verwendete Markierungsstrich hat standardmäßig die Breite 0 mm und fehlt daher. Gegebenenfalls kann er mit Hilfe des Befehls `\footnoterule`² eingefügt werden. Der eingefügte Markierungsstrich darf aber keinen Platz beanspruchen. Dazu ergänzt man den Strich durch korrigierende

negative Abstände.

Die Fußnoten werden in einer kleineren Schrift gesetzt.³

¹Fußnote in der ersten Spalte.

²Ein L^AT_EX-Standardbefehl.

³Diese dritte Fußnote erstreckt sich über zwei Zeilen und zeigt so das Layout längerer Fußnoten.

Nun folgen Fußnoten im zweispaltigen Satz mit dem Paket multicol:

Der Block der Fußnoten in der rechten Spalte wird durch einen Abstand von der eigentlichen Spalte¹ getrennt. Der sonst in L^AT_EX in der Regel verwendete Markierungsstrich hat standardmäßig die Breite 0 mm und fehlt daher. Gegebenenfalls kann er mit Hilfe des Befehls `\footnoterule`²

eingefügt werden. Der eingefügte Markierungsstrich darf aber keinen Platz beanspruchen. Dazu ergänzt man den Strich durch korrigierende negative Abstände.

Die Fußnoten werden in einer kleineren Schrift gesetzt.³

¹Fußnote in der ersten Spalte.

²Ein L^AT_EX-Standardbefehl.

³Diese dritte Fußnote erstreckt sich über zwei Zeilen und zeigt so das Layout längerer Fußnoten.

Das letzte Beispiel zeigt Fußnoten im zweispaltigen Satz mit dem Paket ftnright:

Der Block der Fußnoten in der rechten Spalte wird durch einen Abstand von der eigentlichen Spalte¹ getrennt. ren Schrift gesetzt.³

Der sonst in L^AT_EX in der Regel verwendete Markierungsstrich hat standardmäßig die Breite 0 mm und fehlt daher. Gegebenenfalls kann er mit Hilfe des Befehls `\footnoterule`² eingefügt werden. Der eingefügte Markierungsstrich darf aber keinen Platz beanspruchen. Dazu ergänzt man den Strich durch korrigierende negative Abstände.

Die Fußnoten werden in einer kleine-

1. Fußnote in der ersten Spalte.
2. Ein L^AT_EX-Standardbefehl.
3. Diese dritte Fußnote erstreckt sich über zwei Zeilen und zeigt so das Layout längerer Fußnoten.

Abbildung 9.1: Vergleich von Fußnotenpositionierung im Standardlayout mit multicol und ftnright

9.10 Paket hline

Werden in Tabellen senkrechte und direkt aufeinander folgenden horizontale Linien gesetzt, so sind die vertikalen Linien an Schnittpunkt mit den mehrfachen horizontalen Linien unterbrochen. Dieses Verhalten ist allerdings nicht immer erwünscht. Für den Fall von Doppellinien stellt David Carlisle mit dem Paket `hline` eine Lösung zur Verfügung.

Mit diesem Paket werden vertikale Linien wie üblich als Argument beim Beginn einer `tabular`-Umgebung vereinbart. Die Befehle `\hline` und `\cline` können weiterhin verwendet werden und verhalten sich wie gewohnt. Der neue Befehl `\hhline` ergänzt diese, wobei er die Funktion der beiden Standard-Befehle übernehmen kann, und zusätzlich ermöglicht das Verhalten bei Kreuzungen mit Doppellinien festzulegen. Der Befehl hat die Form

```
\hhline{<Linienform>}
```

mit einem Argument `<Linienform>`, das ähnlich aufgebaut ist wie die Tabellendefinition. Mit diesem Befehl können sowohl einfache als auch doppelte horizontale Linien – auch abschnittsweise – festgelegt werden.

Die Festlegung der Linie und des Verhaltens beim Kreuzen mit vertikalen Doppellinien wird analog zur Tabellendefinition durch Buchstaben und Sonderzeichen festgelegt. Die Zeichen haben die folgende Bedeutung:

Horizontale Linien

- Einfache horizontale Linie über eine einzelne Spalte;
- = Doppelte horizontale Linie über eine einzelne Spalte;
- ~ Spalte ohne horizontale Linie(n).

Vertikale Linienstücke

- | Vertikales Linienstück zwischen den beiden Linien einer horizontalen Doppellinie.
- : Ohne vertikale Verbindung zwischen den beiden Linien einer horizontalen Doppellinie.

Linienstücke bei Kreuzungen horizontaler und vertikaler Linien

- # Verbindungen zwischen den horizontalen und vertikalen Doppellinien.
- t Horizontales Linienstück zwischen den beiden Linien einer vertikalen Doppellinie zur Verbindung der *oberen* Linie einer horizontalen Doppellinie.
- b Wie t, aber zur Verbindung der *unteren* Linie einer horizontalen Doppellinie.

Wiederholungen

- * Verhält sich identisch mit der gleichen Angabe bei der Tabellendefinition und erlaubt das mehrfache Verwenden von Angaben. Die Form ist `*{<Anzahl>}{<Angaben>}`. Die Wirkung ist identisch mit der `<Anzahl>`-fachen Angabe des zweiten Arguments.

Im folgenden Beispiel sieht man die Verwendung der verschiedenen Linienstücke bei Kreuzungen sowie die Unterschiede in den horizontalen Linien bei Verwendung von `\hhline` und `\cline`:

```

7  \usepackage{hhline}
8  \begin{document}
9  \begin{tabular}{|cc|cc|}
10 \hhline{|t==:tt==:t|}
11   A & B & C & D\\
12 \hhline{|:==:tt==:|}
13   a & b & c & d\\
14 \hhline{||-~||-~||}
15   e & f & g & h\\
16 \cline{1-1}\cline{3-4}
17   i & j & k & l\\
18 \hhline{|b==:bb==:b|}
19 \end{tabular}
20 \end{document}

```

A	B	C	D
a	b	c	d
e	f	g	h
i	j	k	l

[Buch1/Ergaenz/hhline1.tex]

Die Angaben zu vertikalen Linienstücken und Linien bei Kreuzungen betreffen nur vorhandene oder fehlende Verbindungen innerhalb des Bereichs einer Doppellinie. Sie haben keinen Einfluss auf Linien, die Spalten trennen. Bei vertikalen Linien muss zur Unterbrechung `:` angegeben werden, da nur so bei einer Doppellinie eine Zuordnung zu einer bestimmten vertikalen Linie möglich ist. Bei horizontalen Linien fehlt in diesem Fall eine Angabe.

Werden in einem `\hhline`-Befehl sowohl einfache als auch doppelte horizontale Linien verwendet, so liegen die einfachen Linien auf gleicher Höhe wie die Untere der Doppellinien.

Die Angabe `#` erzeugt die gleichen Linienstücke wie die Angabe `|tb|`. Allerdings sollte bevorzugt `#` verwendet werden, da diese Form nicht nur in der Eingabe kürzer, sondern auch wesentlich effizienter ist.

Der Abstand der beiden Linien einer Doppellinie wird durch die Länge

```
\doublerulesep
```

festgelegt. Die Länge

```
\arrayrulewidth
```

bestimmt wie üblich die Strichstärke der gesetzten Linien.

Im nächsten Beispiel ist zu sehen, dass auch mehr als zwei vertikale Linien korrekt behandelt werden. Bei horizontalen Mehrfachlinien treten hingegen Probleme mit dem Abstand zwischen direkt aufeinanderfolgenden `\hhline`-Befehlen auf, die sich jedoch mit Hilfe einer Leerzeile und

negativen Abständen korrigieren lassen. Das gilt auch für Kombinationen von `\hline`- mit `\hhline`-Befehlen.

```

9  \begin{tabular}{||c|c|||cc||}
10  \hhline{|t:::t|t:::t|}
11    A & B & C & D\\
12  \hhline{|t:::t|t:::t|}
13    \multicolumn{1}{||c|}{i}
14    & j & k & l\\
15  \hhline{||--|||--||}
16    \multicolumn{1}{||c|}{i}
17    & j & k & l\\
18  \hhline{||--|b|b|--||}
19    m & n & o & p\\
20  \hhline{||--#tb|=#||}
21    q & r &
22    \multicolumn{1}{c|}{s}
23    & t\\
24  \hhline{|:~::~b|b::~:|}
25    \multicolumn{1}{||c|}{W}
26    & \multicolumn{1}{c|}{X}
27    & Y & Z\\
28  \hhline{:tb:::t|t:::tb:}
29    w & x & y & z \\
30  \hhline{|b::~||b::~:b|}
31  \end{tabular}

```

A	B	C	D
i	j	k	l
i	j	k	l
m	n	o	p
q	r	s	t
W	X	Y	Z
w	x	y	z

[Buch1/Ergaenz/hhline2.tex]

Man beachte die unterschiedlichen Abstände der Buchstaben zur nachfolgenden horizontalen Linie in den beiden Zeilen mit `i` in der ersten Spalte. Der größere Abstand in der zweiten Zeile wird durch die Angabe von `b` zur Verbindung der horizontalen Linie zwischen den vertikalen Mehrfachlinien verursacht, da aufgrund dieser Angabe Platz für eine Doppellinie reserviert wird.

Daher sollte das Argument eines `\hhline`-Befehls, der eine einfache horizontale Linie erstellen soll, nur die Angaben `-`, `~` und `|` enthalten. `*`-Ausdrücke können auch verwendet werden, sollten aber als zu wiederholendes Argument ebenfalls nur die genannten Angaben enthalten.

Die horizontalen Linien des Paktes `hhline` werden aus Linienstücken zusammengesetzt während `\hline` eine einzige Linie erzeugt. Obwohl $\text{\LaTeX}$ die Teile sehr exakt positioniert, kann es vorkommen, dass bei der Umsetzung durch einen Druckertreiber die Linienstücke nicht sauber anschließen. Als Abhilfe wird empfohlen einen anderen Druckertreiber zu verwenden, oder – wenn das nicht möglich ist – die Strichstärke (`\arrayrulewidth`) zu erhöhen, um den Effekt zu reduzieren.

```

7 \usepackage{array,hhline}
8 \begin{document}
9 \newlength{\slc}
10 \newlength{\dlc}
11 \setlength{\slc}{\doublerulesep}
12 \addtolength{\slc}{\arrayrulewidth}
13 \setlength{\dlc}{\baselineskip}
14 \addtolength{\dlc}{-\slc}
15 \begin{tabular}{||c|c||cc||}
16 \hhline{||==|t:==:t|}
17 A & B & C & D\\
18 \hhline{|b:==|b:==:b|}
19 \multicolumn{2}{|c|}{@{\hspace{\slc}}{}}{ }
20 & \multicolumn{2}{c|}{ }\backslash[-\dlc]
21 \hhline{|t:==|t:==:t|}
22 \multicolumn{1}{|c|}{i} & j & k & l\\
23 \hhline{||--||--||}
24 \multicolumn{1}{|c|}{i} & j & k & l\backslash[-\slc]
25 \hhline{||--||b|--||}
26 m & n & o & p\\
27 \hhline{|b:==|b:==:b|}
28 \end{tabular}
29 \quad
30 \begin{tabular}{||c|c||cc||}
31 \hline\hline
32 A & B & C & D\\
33 \hline\hline\hline\hline
34 \multicolumn{1}{|c|}{i} & j & k & l\\
35 \hline
36 \multicolumn{1}{|c|}{i} & j & k & l\\
37 \hline
38 m & n & o & p\\
39 \hline\hline
40 \end{tabular}
41 \end{document}

```

A	B	C	D
i	j	k	l
i	j	k	l
m	n	o	p

A	B	C	D
i	j	k	l
i	j	k	l
m	n	o	p

[Buch1/Ergaenz/hhline3.tex]

In diesem Beispiel wird neben dem Paket `hhline` auch das Paket `array` geladen. Das ist für eine korrekte Formatierung erforderlich, da $\text{\LaTeX}$ ohne das Paket `array` Mehrfachlinien in einem Abstand von `\doublerulesep`

der Linienmitten setzt, während `hhline` horizontale Doppellinien mit einem *Zwischenraum* von `\doublerulesep` setzt. Vertikale Mehrfachlinien werden von `hhline` ohne `array` mit dem auf die Linienmitten bezogenen Abstand gesetzt.

Die selbstdefinierten Längen `\slc` und `\dlc` dienen zur Korrektur von Abständen der Linien. `\slc` ist dabei die Summe aus dem Linienabstand `\doublerulesep` und der Strichdicke `\arrayrulewidth` der Linien, also der für eine zusätzliche Linie benötigte Raum. `\dlc` ist der (hier normale) Zeilenabstand vermindert um den für eine zusätzliche Linie benötigten Raum.

Die beiden `\multicolumn`-Befehle in den Zeilen 19 und 20 dienen dazu, die vertikalen Linien an den gewünschten Stellen zu verbinden. Der `@`-Ausdruck im ersten `\multicolumn` hat den Zweck, die vertikalen Linien in der Mitte der Zeile korrekt zu positionieren, indem an der Stelle, wo eine Linie unterbrochen werden soll, der dafür benötigte Platz durch Angabe von `@{\hspace{\slc}}` eingefügt wird. Zur Korrektur des durch die Zeile verursachten Abstands wird beim Zeilenumbruch ein zusätzlicher Vorschub von `-\dlc` (Zeilenabstand – Raum für eine zusätzliche Linie) verlangt.

Auch Zeile 24 bedarf einer Korrektur. Sie ist erforderlich, da der `\hhline`-Befehl in der folgenden Zeile eine einzelne Linie setzen soll, die auch zwischen einigen – aber nicht allen – vertikalen Linien verlaufen soll und dazu `b` verwendet wird. Durch die Verwendung von `b` wird jedoch Platz für eine Doppel- statt für eine einfache Linie reserviert, was die Korrektur erforderlich macht.

9.11 Paket indentfirst

Auch dieses kleine Hilfspaket wird von David Carlisle zur Verfügung gestellt. Es bietet eine Alternative zum Standardverhalten von $\text{\LaTeX}$, dass die erste Zeile des ersten Absatzes nach einer Überschrift nicht eingerückt wird, auch wenn bei allen anderen Absätzen so verfahren wird. Wird das Paket `indentfirst` geladen, rückt $\text{\LaTeX}$ auch die erste Zeile des ersten Absatzes nach einer Überschrift ein. Zusätzliche Befehle oder Umgebungen stellt das Paket nicht zur Verfügung.

Das unterschiedliche Verhalten zeigen die beiden folgenden Beispiele.

```

1 \documentclass{article}
2 \usepackage[T1]{fontenc}
3 \usepackage[ngerman]{babel}
4 \setlength{\textwidth}{5cm}
5 \pagestyle{empty}
6 \begin{document}
7 \section*{Standard-\LaTeX}
8
9 Der erste Absatz nach einer
10 "Überschrift wird im Gegensatz
11 zu anderen Abs"atzen nicht
12 eingezogen.
13
14 Die erste Zeile der anderen
15 Abs"atze wird einger"uckt.
16 \end{document}

```

Standard- $\text{\LaTeX}$

Der erste Absatz nach einer Überschrift wird im Gegensatz zu anderen Absätzen nicht eingezogen.

Die erste Zeile der anderen Absätze wird eingerückt.

[Buch1/Ergaenz/indentfirst0.tex]

```

1 \documentclass{article}
2 \usepackage[T1]{fontenc}
3 \usepackage[ngerman]{babel}
4 \usepackage{indentfirst}
5 \pagestyle{empty}
6 \setlength{\textwidth}{5cm}
7 \pagestyle{empty}
8 \begin{document}
9 \section*{Indentfirst}
10
11 Auch der erste Absatz nach
12 einer "Überschrift wird
13 eingezogen.
14
15 Die erste Zeile der anderen
16 Abs"atze wird ebenfalls
17 einger"uckt.
18 \end{document}

```

Indentfirst

Auch der erste Absatz nach einer Überschrift wird eingezogen.

Die erste Zeile der anderen Absätze wird ebenfalls eingerückt.

[Buch1/Ergaenz/indentfirst.tex]

9.12 Paket layout

Dieses Paket von Kent McPherson erlaubt es, das Layout eines Dokuments zu überprüfen. Johannes Braams stellte das Paket auf $\text{\LaTeX 2}_{\epsilon}$ um, und Hideo Umeki nahm einige Modifikationen vor. Das Paket wird geladen mit dem Befehl

```
\usepackage[<Option(en)>]{layout}
```

und unterstützt die Ausgabe in mehreren Sprachen, die als Optionen angegeben werden können.

Die Beschriftung der grafischen Darstellung der Seite im Maßstab 1:2 unter Berücksichtigung des zu Grunde liegenden Papierformats enthält eine Reihe von Bezeichnungen, die durch Angabe einer der folgenden Sprachoptionen angepasst werden können:

dutch, german, ngerman, english, french, francais, spanish, portuguese, brazilian, italian.

Voreingestellt ist die Ausgabe der Bezeichnungen in Englisch.

Weitere Option legen fest, mit welcher Genauigkeit die Längen ausgegeben werden, und ob die wichtigsten Längen auch am Bildschirm gelistet sowie in der `.log`-Datei nachgelesen werden können:

verbose	Ausgabe von <code>\hoffset</code> , <code>\voffset</code> , <code>\textheight</code> sowie <code>\textwidth</code> am Bildschirm und in die <code>.log</code> -Datei.
silent	Unterdrücken der Ausgabe der Längen am Bildschirm und in die <code>.log</code> -Datei. (Voreinstellung)
integers	Ausgabe der Längen in der Maßeinheit <code>pt</code> als ganzzahliger Werte (Voreinstellung)
reals	Ausgabe der Längen in der Maßeinheit <code>pt</code> mit bis zu fünf Nachkommastellen für höhere Genauigkeit

Das Paket unterscheidet zwischen einseitigen und zweiseitigen Layouts. Bei zweiseitigem Layout werden sowohl gerade als auch ungerade Seiten getrennt grafisch unter Angabe der verwendeten Längen dargestellt. Die Ausgabe des Layouts wird veranlasst durch einen der beiden folgenden Befehle:

```
\layout
\layout*
```

Die erste Form zeigt das Layout, wie es bei `\begin{document}` eingestellt ist. Später vorgenommene Änderungen werden nicht berücksichtigt. Die zweite Form `\layout*` bewirkt die Ausgabe des aktuellen Layouts nach Änderungen. Folgende `\layout`-Befehle zeigen die geänderten Werte, wie sie beim letzten `\layout*` gültig waren.

Der Aufbau der grafischen Darstellung erfolgt größtenteils mit Befehlen, die auch dem Anwender zugänglich sind. Auf eine Beschreibung dieser Befehle wird hier aber weitgehend verzichtet. Für zusätzliche sprachliche Anpassungen werden hier nur diejenigen Befehle vorgestellt, die für Bezeichnungen verwendet werden:

`\Headertext{<Text>}`

Bezeichnung für die Kopfzeile.

`\Bodytext{<Text>}`

Bezeichnung für den Haupttext.

`\Footertext{<Text>}`

Bezeichnung für die Fußzeile.

`\MarginNotestext{<Text>}`

Bezeichnung für die Randnotizen.

`\oneinchttext{<Text>}`

Bezeichnung für die Maßeinheit Inch (ein Zoll).

`\notshown{<Text>}`

Text für nicht abgebildete Längen (ohne Abbildung).

Verwendet werden die für Deutsch festgelegten Bezeichnungen oder der im Deutschen gültige Wert.

9.13 Paket longtable

Tabellen sind in $\text{\LaTeX}$ auf eine einzelne Seite beschränkt. Eine Fortsetzung auf Folgeseiten ist nicht vorgesehen. Das Paket `longtable` von David Carlisle löst das Problem langer Tabellen durch die Umgebung `longtable`, in der Seitenumbrüche zugelassen sind oder sogar explizit vorgegeben werden können. In dieser Umgebung können Kopf- und Fußzeilen verwendet werden, die auf jeder Seite der Tabelle wiederholt werden. Die Kopfzeilen der ersten Seite und die Fußzeilen auf der letzten Seite der

Tabelle können getrennt angegeben werden und treten dann an die Stelle der normalen Kopf- oder Fußzeilen.

Gleichzeitig hat die Umgebung `longtable` Eigenschaften einer `table`-Umgebung. So steht in ihr ein `\caption`-Befehl zur Verfügung, der in Kopf- oder Fußzeilen verwendet werden kann. Neben dem aus der `table`-Umgebung bekannten `\caption`-Befehl, der sich bei `longtable` völlig gleich verhält, stehen auch Varianten zur Verfügung, die lediglich eine Überschrift hinzufügen, aber keine Tabellenummer vergeben und auch keinen Eintrag in das Tabellenverzeichnis vornehmen.

Die Pakete `array`, `dcolumn` und `hhline` können wie bei einer `tabular`-Umgebung genutzt werden. Damit stehen bei `longtable` die gleichen Erweiterungen zur Verfügung wie bei `tabular`.

Das Setzen der Tabelle erfolgt durch mehrere $\text{\LaTeX}$ -Aufrufe. Hintergrund ist, dass die Tabelle stückweise aufgebaut wird, und die Spaltenbreiten auf diesen Teiltabellen basieren. Über die `.aux`-Datei werden die Informationen über Spaltenbreiten an den nächsten $\text{\LaTeX}$ -Aufruf weitergegeben. Die Anzahl der notwendigen Aufrufe hängt dabei ab von der – einstellbaren – Größe der Teiltabellen und der Komplexität der Gesamttabelle. Mehrspaltige Einträge in der Tabelle, die Einfluss auf die Spaltenbreiten haben, machen dabei zusätzliche Aufrufe erforderlich. Eine allgemeine Aussage über die Anzahl der $\text{\LaTeX}$ -Aufrufe, die zur korrekten Ausgabe der Tabelle führen, ist daher nicht möglich, meist genügen jedoch wenige Aufrufe.

Das Paket `longtable` wird wie üblich mit dem `\usepackage`-Befehl geladen:

```
\usepackage[<Option(en)>]{longtable}
```

Die folgenden Optionen werden unterstützt:

- errorshow** Unterdrückt die Ausgabe von Warnungen, nur Fehler werden ausgegeben.
- pausing** Behandelt Warnungen wie Fehler. Daher hält $\text{\TeX}$ auch bei Warnungen an.
- set | final** Optionen ohne Wirkung, nur zur Kompatibilität mit früheren Versionen, in denen sie einem Befehl `\setlongtables` entsprachen.

Wie üblich können die Optionen auch global in der `\documentclass`-Anweisung angegeben werden. Das Paket stellt eine Umgebung `longtable` zur Verfügung, die abgesehen von den optionalen Angaben zum Ausrichten der gesamten Tabelle, exakt der `tabular`-Umgebung entspricht. Eine Einschränkung besteht darin, dass die Umgebung nur im einspaltigen – und nicht im mehrspaltigen – Satz verwendet werden kann.

```
\begin{longtable}[<Ausrichtung>]{<Spaltendefinition>
  <Festlegung der Kopf- und Fußzeilen>
  <Tabellenzeilen>
\end{longtable}
```

Ein Unterschied zur `tabular`-Umgebung ist die Angabe zur *<Ausrichtung>*. Ein `tabular` wird wie ein einzelnes Zeichen im Text positioniert, und die Angaben legen die vertikale Positionierung gegenüber der Grundlinie fest. `longtable` wird hingegen ähnlich wie eine `table` positioniert und vom Text abgesetzt. Die vertikale Ausrichtung gegenüber der Grundlinie spielt daher keine Rolle. Da die Tabelle in der Regel schmaler ist als die Textbreite, dient bei `longtable` das Argument *<Ausrichtung>* zur Angabe der horizontalen Ausrichtung in der Zeile.

- c** Tabelle horizontal zentrieren (default);
- l** Tabelle linksbündig setzen;
- r** Tabelle rechtsbündig setzen.

Bei fehlender Angabe wird die Ausrichtung mit Hilfe der beiden Längen `\LTleft` und `\LTRight` festgelegt. Sie sind so voreingestellt, dass die Tabelle zentriert wird. Durch verändern der beiden Längen kann beispielsweise ein fester Abstand vom Zeilenanfang oder -ende vorgegeben werden.

Keinerlei Unterschiede zur `tabular`-Umgebung gibt es bei der Vereinbarung der Spalten im Argument *<Spaltendefinition>*. Auch spezielle Angaben wie `@{\extracolsep{...}}` wie bei `tabular*` können genutzt werden, wobei die Festlegung der Tabellenbreite aber durch Vorgabe fester Abstände vom linken und rechten Rand erfolgt (nicht bei `\begin{longtable}`). Auf eine Beschreibung der Angaben kann daher hier verzichtet werden.

Auch die Erweiterungen der Pakete `array` und `dcolumn` können in Verbindung mit `longtable` genutzt werden, sofern sie geladen sind. Auch beim Inhalt der Tabelle, oben als *<Tabellenzeilen>* angedeutet, gibt es keine Unterschiede. `\multicolumn`-Befehle können wie üblich verwendet werden. Allerdings können in einer `longtable`-Umgebung im Gegensatz zu einer `tabular`-Umgebung zwischen Tabellenzeilen – genauer am Anfang

einer neuen Tabellenzeile – Befehle zum Seitenumbruch eingeschoben werden.

Neu hinzugekommen sind Kopf- und Fußzeilen. Unterschieden wird hierbei zwischen Kopfzeilen am Anfang der Tabelle und Kopfzeilen *nach* einem Seitenumbruch innerhalb der Tabelle. Analog wird unterschieden zwischen Fußzeilen *vor* einem Seitenumbruch in der Tabelle und den Abschlusszeilen der Tabelle. Die Kopf- und Fußzeilen sind gewöhnliche Tabellenzeilen oder eine Tabellenzeile, die nur einen `\caption`-Befehl enthält. Der im folgenden genauer beschriebene `\caption`-Befehl besteht dabei aus einem `\multicolumn`-Befehl über alle Spalten der Tabelle.

Zur Beschleunigung des Setzens und zur Vermeidung zusätzlicher Formatierläufe bietet die `longtable`-Umgebung die Möglichkeit, wie bei der `tabbing`-Umgebung Beispielzeilen anzugeben, die die Spaltenbreiten (mit)bestimmen, aber nicht gesetzt werden. Wie bei der `tabbing`-Umgebung werden diese Beispielzeilen statt mit `\\` mit dem Befehl

```
\kill
```

abgeschlossen. `\kill`-Zeilen können sowohl in Kopf- und Fußzeilen wie auch innerhalb der Tabelle verwendet werden. Für den korrekten Satz der Tabelle sind `\kill`-Zeilen allerdings selbst in komplizierten Fällen – insbesondere im Fall mehrfacher `\multicolumn`-Befehle, die Einfluss auf die Spaltenbreiten haben – nicht erforderlich. Ohne sie werden allenfalls zusätzliche Formatierungsaufrufe notwendig. Auf die Notwendigkeit weiterer Aufrufe wird beim Formatieren und in der `.log`-Datei mit der Meldung

```
Package longtable Warning: Table widths have changed.
Rerun LaTeX.
```

hingewiesen.

Die Tabellenzeilen – oben als *<Tabellezeilen>* bezeichnet – haben die von der `tabular`-Umgebung gewohnte Form. `\multicolumn`-Befehle können ebenso verwendet werden wie `\hline`- und `\cline`-Befehle oder Varianten aus dem `hhline`-Paket. Gegebenenfalls kann auch das optionale Argument des `\\`-Befehls genutzt werden, um den einer Zeile folgenden Abstand zu verändern. Für den beim Paket `array` beschriebenen Sonderfall einer Schlussspalte mit Zeilenumbruch, die mit `\centering`, `\raggedright` oder `\raggedleft` gesetzt wird, stellt auch `longtable` den Befehl `\tabularnewline` zur Verfügung.

Bisher wurde die Grundform der Umgebung beschrieben. Tabellen, die am Ende einer Seite umbrochen und auf der nächsten Seite fortgesetzt werden, sind allein durch den Wechsel von einer `tabular`- zu einer `longtable`-Umgebung möglich. Bei Tabellen mit Spaltenüberschriften wird man jedoch in der Regel die Spaltenüberschriften bei den getrennt gesetzten Teilen der Tabelle wiederholen wollen.

Die Kopf- und Fußzeilen der Tabelle werden am Anfang der Tabelle festgelegt. Diese Zeilen sind normale Tabellenzeilen, wie sie auch innerhalb der Tabelle vorkommen können. Die Kopf- und Fußzeilen werden jeweils durch einen Befehl abgeschlossen, der die Funktion der vorhergehenden Zeilen festlegt. Unterschieden wird dabei zwischen den auf jeder Seite wiederholten Kopf- und Fußzeilen und den Kopfzeilen am Anfang sowie den Fußzeilen am Ende der Tabelle, die an die Stelle der wiederholten Kopf- und Fußzeilen treten, wenn sie angegeben sind. Die vier Befehle zum Abschluss der jeweiligen Angaben lauten:

<code>\endhead</code>	Kopfzeilen nach einem Seitenumbruch
<code>\endfoot</code>	Fußzeilen vor einem Seitenumbruch
<code>\endfirsthead</code>	Kopfzeilen am Tabellenanfang
<code>\endlastfoot</code>	Fußzeilen am Tabellenende

Die Kopf- und Fußzeilen sind direkt nach `\begin{longtable}` anzugeben. Die Reihenfolge der Angaben spielt keine Rolle. Kopf- und Fußzeilen sind normale Tabellenzeilen – jeweils mit `\` abgeschlossen – und werden mit einem der vier Befehle abgeschlossen. Ein `\` direkt vor einem der vier Befehle kann entfallen. Der Abschlussbefehl legt fest, dass alle Zeilen zwischen `\begin{longtable}` oder einem vorhergehenden Abschlussbefehl in der durch den Befehl festgelegten Form verwendet werden.

Speziell für Kopf- und Fußzeilen ist der `\caption`-Befehl vorgesehen, der die gleiche Funktion wie in einer `table`-Umgebung hat. Technisch gesehen handelt es sich um einen `\multicolumn`-Befehl über alle Spalten der Tabelle, der die Tabellenüberschrift zentriert in einer `\parbox` der Breite `\LTcapwidth` (voreingestellt: 4 in) setzt. Ein `\caption`-Befehl ist wie jede andere Tabellenzeile mit `\` abzuschließen. Er hat die folgende Form:

```
\caption[<Kurztitel>]{<Titel>}
\caption*{<Titel>}
```

Die erste Form setzt das Wort *Tabelle* gefolgt von Tabellennummer, einem Doppelpunkt und `<Titel>`. Die Nummerierung erfolgt gemeinsam mit der `table`-Umgebung. Abweichend von der `table`-Umgebung führt

bei `longtable` eine Wiederholung des `\caption`-Befehls innerhalb der Umgebung nicht zur Vergabe einer neuen Tabellennummer. Damit wird sichergestellt, dass er auch in den wiederholten Kopf- und Fußzeilen verwendet werden kann. Fehlt das optionale Argument, wird `<Titel>` auch in das Tabellenverzeichnis eingetragen. Ist hingegen `<Kurztitel>` angegeben, wird dieser statt dem Titel in das Tabellenverzeichnis aufgenommen.

Bei `\caption`-Befehlen in wiederholten Kopf- oder Fußzeilen ist ein wiederholter Eintrag (für jede Seite) in der Regel nicht erwünscht, obwohl eine Wiederholung der Tabellennummer durchaus Sinn macht. Für diesen Fall gibt es die Sonderform, dass ein leeres optionales Argument `<Kurztitel>` angegeben wird, das demzufolge die Form `[]` hat. Bei dieser Sonderform müssen die öffnende und die schließende eckige Klammer direkt aufeinander folgen. Es erfolgt kein Eintrag ins Tabellenverzeichnis. Das Layout des `\caption`-Befehls kann übrigens wie bei anderen Gleitobjekten mit Hilfe des Pakets `caption`, das hier nicht beschrieben wird, verändert werden.

Diese erste Form unterstützt auch `\label`-Befehle. Zu beachten ist allerdings, dass er nicht mehrfach verwendet wird und daher in wiederholten Kopf- und Fußzeilen nicht genutzt werden sollte. Ein `\label`-Befehl sollte daher nur zusammen mit einem `\caption`-Befehl in den Kopfzeilen am Tabellenanfang oder den Fußzeilen am Tabellenende stehen. Ein `\label`-Befehl sollte allerdings keinesfalls der erste Befehl der Kopf- oder Fußzeilen sein. Es empfiehlt sich, ihn an den zugehörigen `\caption`-Befehl anzuhängen.

Die zweite Form setzt im Unterschied zur ersten nur `<Titel>`. Die von der ersten Form hinzugefügte Bezeichnung *Tabelle* und die Tabellennummer fehlen. Ein Eintrag im Tabellenverzeichnis unterbleibt ebenfalls. Diese Form des `\caption`-Befehls eignet sich daher sehr gut für Zwischenüberschriften in der Tabelle.

Der wichtigste Unterschied zwischen `longtable` und `tabular` besteht darin, dass eine `longtable` auf Folgeseiten fortgesetzt werden kann, wenn sie nicht mehr auf die aktuelle Seite passt. Neben automatischen Seitenumbrüchen können Seitenumbrüche auch explizit mit `\pagebreak`- oder `\newpage`-Befehlen vorgenommen oder mit `\nopagebreak` unterdrückt werden. Wie gewohnt kann bei `\pagebreak` und `\nopagebreak` ein optionales Argument angegeben werden, um die Dringlichkeit vorzugeben. Die Befehle zum Seitenumbruch müssen jeweils am Anfang einer Tabellenzeile stehen. Ein Seitenumbruch zwischen zwei Tabellenzeilen kann wie üblich auch dadurch verhindert werden, dass die erste Zeile mit `\\*` abgeschlossen wird.

Ein weiterer Unterschied ist, dass in `longtable` Fußnoten zugelassen sind, die bei `tabular` nur möglich sind, wenn `tabular` in einer `mini-page`-Umgebung steht. Allerdings kann der `\footnote`-Befehl nicht in Kopf- und Fußzeilen der Tabelle verwendet werden, statt dessen kommt der Befehl `\footnotemark` zum Einsatz. Der zugehörige `\footnotetext`-Befehl ist innerhalb der normalen Tabellenzeilen anzugeben. Innerhalb eines `\caption`-Befehls muss der `\footnotemark`-Befehl mit `\protect` geschützt werden.

Das horizontale Ausrichten der Tabelle auf der Seite kann in den Standardfällen – zentriert, links- oder rechtsbündig – durch Angabe eines optionalen Arguments erfolgen. Eine feinere Steuerung ist mit Hilfe von Längenbefehlen möglich. Weitere Längen legen die Abstände vor und hinter der Tabelle und die Breite der Tabellenüber- und -unterschriften fest. Das Paket `longtable` verwendet dazu die folgenden Längen:

<code>\LTleft</code>	Abstand der Tabelle vom linken Rand; (voreingestellt: <code>\fill</code>)
<code>\LTright</code>	Abstand der Tabelle vom rechten Rand; (voreingestellt: <code>\fill</code>)
<code>\LTpre</code>	Abstand der Tabelle vom vorhergehenden Text; (voreingestellt: <code>\bigskipamount</code>)
<code>\LTpost</code>	Abstand der Tabelle zum nachfolgenden Text; (voreingestellt: <code>\bigskipamount</code>)
<code>\LTcapwidth</code>	Breite der <code>\parbox</code> , in der Tabellenüber- und -unterschriften gesetzt werden. (voreingestellt: 4 in)

Die beiden Längen `\LTleft` und `\LTright` dienen auch dazu, eine `longtable` einer vorgegebenen Breite zu setzen, wie es `tabular*` bei einer normalen Tabelle erlaubt. Der einfachste Fall ist eine Tabelle der Breite `\textwidth`, bei der beide Längen auf 0 mm gesetzt werden müssen. Bei einer festen vorgegebenen Tabellenbreite müssen die beiden Längen hingegen berechnet werden. Zum Setzen einer dreispaltigen, auf der Seite zentrierten Tabelle der Breite 8 cm könnte etwa der folgende Code verwendet werden:

```
\setlength{\LTleft}{\textwidth}
\addtolength{\LTleft}{-8cm}
\setlength{\LTleft}{.5\LTleft}
\setlength{\LTright}{\LTleft}
\begin{longtable}{c*{2}{@{\extracolsep{\fill}}c}}
...
\end{longtable}
```

Dabei ist angenommen, dass die Textbreite mindestens 8 cm beträgt. Andernfalls ragt die `longtable` links und rechts gleich weit über die Zeile hinaus.

Der Aufbau der fertigen Tabelle erfolgt im Rahmen mehrfacher Formatierungsaufrufe. Dabei wird die Tabelle unter Verwendung bereits verfügbarer Informationen über Spaltenbreiten abschnittsweise aufgebaut. $\text{\TeX}$ muss den kompletten Abschnitt in seinem Speicher halten. Bei sehr großen Abschnitten – etwa einer kompletten vielseitigen Tabelle – kann es daher vorkommen, dass $\text{\TeX}$ wegen fehlendem Speicher die Formatierung abbrechen muss. Das Problem wird dadurch umgangen, dass das Paket den Zähler

```
LTchunksize
```

verwendet, um die Anzahl der Tabellenzeilen eines solchen Abschnitts festzulegen. Voreingestellt ist ein Wert von 20 Tabellenzeilen. Das Erhöhen des Werts kann gegebenenfalls Formatierungsaufrufe sparen. Allerdings erhöht sich dabei der Speicherbedarf.

Ohne Einfluss auf die fertige Tabelle ist auch der Befehl `\setlongtables`, der keinerlei Funktion hat, aber in älteren Versionen des Pakets eine Rolle spielte und daher aus Kompatibilitätsgründen weiterhin definiert ist.

9.14 Paket multicol

$\text{\LaTeX} 2_{\epsilon}$ unterstützt neben der (normalen) einspaltigen Ausgabe auch einen zweispaltigen Satz von Dokumenten. Für den zweispaltigen Druck ist die Option `twocolumn` anzugeben. Der Satz in drei oder mehr Spalten wird von $\text{\LaTeX} 2_{\epsilon}$ selbst nicht unterstützt.

Abhilfe schafft das Paket `multicol` von Frank Mittelbach, das bis zu 10 Spalten setzen kann. Im Gegensatz zum zweispaltigen Druck mit Hilfe der Option `twocolumn` wird bei einer aktuellen Änderung der Spaltenanzahl eine neue Seite nur begonnen, wenn nicht mehr ausreichend Platz auf der Seite übrig ist. Im Zusammenhang damit werden auch die auf der letzten Seite gedruckten Spalten ausbalanciert, damit sie (ungefähr) die gleiche Länge haben. Fußnoten werden am Ende der Seite *einspaltig* gesetzt – alternativ kann man das Paket `fnright` verwenden, um die Fußnoten einer Seite am Ende der am weitesten rechts stehenden Spalte der Seite zu setzen.

Eingeschränkt ist allerdings die Verwendung von Gleitobjekten und Randnotizen: In mehrspaltigen Teilen des Dokuments können nur Gleitobjekte verwendet werden, die sich über die gesamte Breite der Seite erstrecken. Andere Gleitobjekte wie `table` oder `figure` werden ignoriert und daher – ohne Warnung oder Fehlermeldung – auch nicht gesetzt. Die zugelassenen Gleitobjekte werden stets am Seitenanfang, am Seitenende oder auf einer für Gleitobjekte reservierten Seite ausgegeben; eine Positionierung an der Stelle der Vereinbarung im Quelltext ist nicht möglich. Randnotizen mit `\marginpar` sind im Mehrspaltensatz mit `multicols` nicht möglich.

Zum Laden des Pakets wird der Befehl

```
\usepackage[<Debug>]{multicol}
```

verwendet. Durch Angabe von Optionen lässt sich steuern, in welchem Umfang das Paket beim Formatieren mehrerer Spalten Meldungen ausgibt. In der Regel wird `multicol` daher ohne das optionale Argument geladen. Die folgenden Debug-Optionen sind vorgesehen:

grid	Diese Option ist für Rastersatz vorgesehen, aktuell bewirkt sie allerdings nur eine Warnung, wenn das Paket den Eindruck hat, dass das Raster nicht eingehalten wurde.
errorshow	Unterdrückt alle Ausgaben außer Fehlermeldungen (Voreinstellung);
infoshow	Ausgabe von Informationen über den Beginn und das Ende einer <code>multicols</code> -Umgebung, die Breite der Spalten und den verfügbaren Platz auf der Seite beim Beginn und am Ende der Umgebung. Die Meldungen werden jeweils mit <code>Package multicol:</code> eingeleitet, Folgezeilen mit <code>(multicol)</code> .
balancingshow	Zusätzlich zu den Informationen von <code>infoshow</code> erfolgt die Ausgabe der Höhe der Spalten und ihrer Qualität (engl. <i>badness</i>) einschließlich der untersuchten und später verworfenen Varianten.
markshow	Ausgabe von Informationen über ‚Marks‘ zusätzlich zu den Meldungen von <code>balancingshow</code> .
debugshow	Verhält sich wie <code>markshow</code> , erzeugt aber eine zusätzliche horizontale Linie (<code>\hrule</code>) im gesetzten Dokument nach der Zeile, bis zu der beim Seitenumbruch voraus geschaut wurde.

Die eigentliche Steuerung erfolgt durch den auf Seite 285 beschriebenen Zähler `tracingmulticols`, der auch direkt verändert werden kann. Durch Veränderung des Zählerwerts lässt sich der Umfang der Meldungen für einzelne Bereiche lokal verändern.

Für mehrspaltige Bereiche des Dokuments wird die Umgebung `multicols` verwendet:

```
\begin{multicols}{<Spzahl>}[<Einleitung>][<Platz>]
  <Mehrspaltiger Text>
\end{multicols}
```

Das erste Argument `<Spzahl>` legt die Anzahl der zu setzenden Spalten fest und muss eine ganze Zahl zwischen 1 und 10 (einschließlich) sein. Die Anzahl der Spalten sollte allerdings so niedrig wie möglich gewählt werden, da mehr Spalten eine geringere Textbreite in den einzelnen Spalten bedeuten und daher ein vernünftiger Zeilenumbruch immer schwieriger wird.

Das optionale Argument `<Einleitung>` gibt eine Textpassage vor, die vor Beginn des mehrspaltigen Satzes gesetzt wird, beispielsweise eine Überschrift und ein einleitender Absatz. Leerzeilen oder Befehle, um einen neuen Absatz zu beginnen, sind in diesem Argument nicht zugelassen. Es können jedoch mehrere Überschriften direkt gefolgt jeweils von einem einzelnen Absatz angegeben werden. Das Argument kann zwar eine `minipage` enthalten, aber sie unterliegt den gleichen Einschränkungen wie das Argument selbst.

Durch Angabe einer Länge `<Platz>` als zweites optionales Argument kann man festlegen, dass die `multicols`-Umgebung auf einer Seite nur dann beginnen soll, wenn auf der Seite noch mindestens der verlangte Platz frei ist. Andernfalls wird die aktuelle Seite beendet, und `multicols` beginnt erst auf der neuen Seite.

Ist ein Ausgleich der Spalten nicht erwünscht, kann statt der Umgebung `multicols` die Umgebung `multicols*` verwendet werden. Sie wird in der gleichen Form verwendet wie die `multicols`-Umgebung:

```
\begin{multicols*}{<Spzahl>}[<Einleitung>][<Platz>]
  <Mehrspaltiger Text>
\end{multicols*}
```

Die Argumente haben die gleiche Bedeutung wie bei `multicols`. Der einzige Unterschied ist, dass bei einem sich über mehrere Seiten erstreckenden Satz ein Ausgleich der Spalten auf der letzten Seite unterbleibt. Diese letzte Seite wird komplett mehrspaltig gesetzt. Der gesamte Text

befindet sich in den am weitesten links stehenden Spalten, ungenutzte Spalten bleiben leer. Der der Umgebung folgende Text beginnt auf einer neuen Seite.

Die `multicols`-Umgebung kann in vielen anderen Umgebungen sowie in Boxen mit Zeilenumbruch verwendet werden. Dazu gehören insbesondere Listen- und Zitat-Umgebungen sowie Gleitobjekte. Umgekehrt können die meisten Umgebungen auch im mehrspaltigen Satz mit `multicols` genutzt werden. Ausnahmen sind – wie bereits oben erwähnt – die Gleitobjekte, die bei zweispaltigem Satz mit dem `\twocolumn`-Befehl innerhalb der Spalte gesetzt werden. Die Gleitobjekte `figure*` und `table*`, die die gesamte Breite der Seite nutzen, werden jedoch unterstützt. Gleitobjekte erscheinen allerdings frühestens am Anfang der ihrer Vereinbarung folgenden Seite. Die Positionierungsangabe `h` ist im Gegensatz zu `tbp` nicht zugelassen und wird (ohne Warnung) durch `p` ersetzt.

Auch die Schachtelung von `multicols`- und `multicols*`-Umgebungen ineinander ist möglich, wobei eine innere `multicols*`-Umgebung als `multicols`-Umgebung gesetzt und eine Warnung ausgegeben wird. Die Spalte, die die innere `multicols`-Umgebung enthält, wird weiter unterteilt in angegebene Anzahl von Spalten.

Zu beachten ist, dass die innere `multicols`-Umgebung sich nicht über mehrere Spalten der äußeren `multicols`-Umgebung erstrecken kann. Ist sie zu lang, ragt sie über das Seitenende hinaus, und außerdem können Probleme mit dem Ausgleich der Spalten auftreten.

Ein Beispiel für ein Dokument mit geschachteltem Mehrspaltensatz zeigen die beiden Abbildungen auf den nächsten Seiten. In diesem Beispiel wurde das Paket `multicol` mit der Option `debugshow` geladen, um zu zeigen, welche Zeilen der zweiten Seite vor dem endgültigen Zeilenumbruch berücksichtigt wurden. Die einzige andere Abweichung von den Voreinstellungen besteht darin, dass die Spalten durch eine vertikale Linie getrennt sind. Dazu wurde die Länge `\columnseprule` auf den Wert 4 pt gesetzt.

```

1 \documentclass{article}
2 \usepackage[german]{babel}
3 \usepackage[debugshow]{multicol}
4 \usepackage{times}
5 \setlength{\textwidth}{11cm}
6 \setlength{\textheight}{15.5cm}
7 \setlength{\columnseprule}{.4pt}
8
9 \begin{document}
10 \section*{Beispiel zum Paket \textsf{multicol}}
11 %
12 Dieses Beispiel soll das Verhalten des Pakets zeigen.
```

```

13 In der Pr"aambel stehen die beiden folgenden Zeilen,
14 die sich auf das Paket beziehen:
15 %
16 \begin{verbatim}
17   \usepackage[debugshow]{multicol}
18   \setlength{\columnseprule}{.4pt}
19 \end{verbatim}
20 %
21 Es wurde mit der Option \texttt{debugshow} geladen,
22 um die Vorausschau beim Seitenumbruch zu zeigen.
23 Die L"ange \verb.\columnseprule. wurde gesetzt,
24 um die Spalten durch eine vertikale Linie zu trennen.
25
26 \begin{multicols}{2}[%
27   \subsection*{Die
28     Umgebung
29     \texttt{multicols}}
30   %
31   Eine
32   \texttt{multicols}-Umgebung kann "Uberschriften und
33   einleitenden Text enthalten, der "uber alle Spalten
34   gesetzt wird.
35 ]
36 Die Umgebung wurde gesetzt mit den folgenden Befehlen:
37 %
38 \begin{verbatim}
39   \begin{multicols}{2}[%
40     \subsection*{Die
41       Umgebung
42       \texttt{multicols}}
43     %
44     Eine
45     ...
46     Spalten gesetzt wird.
47   ]
48   Die Umgebung wurde ...
49
50   \end{multicols}
51 \end{verbatim}
52 %
53 Andere Einstellungen wurden nicht ver"andert.
54
55 Die Umgebungen \texttt{table} und \texttt{figure}
56 k"onnen hier im Gegensatz zum zweispaltigen Standardsatz
57 mit \verb.\twocolumn. nicht verwendet werden, da sie
58 eigentlich innerhalb der Spalte gesetzt werden m"ussten.
59 M"oglich ist hingegen das Setzen von Abbildungen und
60 Tabellen wie Abbildung-\ref{fig} "uber die gesamte Seite
61 mit den Umgebungen \texttt{figure*} und \texttt{table*}.

```

Beispiel zum Paket multicol

Dieses Beispiel soll das Verhalten des Pakets zeigen. In der Präambel stehen die beiden folgenden Zeilen, die sich auf das Paket beziehen:

```
\usepackage[debugshow]{multicol}
\setlength{\columnseprule}{.4pt}
```

Es wurde mit der Option `debugshow` geladen, um die Vorausschau beim Seitenumbruch zu zeigen. Die Länge `\columnseprule` wurde gesetzt, um die Spalten durch eine vertikale Linie zu trennen.

Die Umgebung multicol

Eine `multicol`-Umgebung kann Überschriften und einleitenden Text enthalten, der über alle Spalten gesetzt wird.

Die Umgebung wurde gesetzt mit den folgenden Befehlen:

```
\begin{multicols}{2}[%
  \subsection*{Die
    Umgebung
    \texttt{multicols}}]
%
Eine
...
Spalten gesetzt wird.
]
Die Umgebung wurde ...

\end{multicols}
```

Andere Einstellungen wurden nicht verändert.

Die Umgebungen `table` und `figure` können hier im Gegensatz zum zweispaltigen Standardsatz

mit `\twocolumn` nicht verwendet werden, da sie eigentlich innerhalb der Spalte gesetzt werden müssten. Möglich ist hingegen das Setzen von Abbildungen und Tabellen wie Abbildung 1 über die gesamte Seite mit den Umgebungen `figure*` und `table*`.

In dieser Umgebung können mathematische Formeln in der üblichen Weise verwendet werden, wie die folgende abgesetzte Formel zeigt:

$$c^2 = a^2 + b^2$$

Die Umgebung `eqnarray` mit den Formeln (1)–(4) zeigt, dass sogar mehrzeilige Formeln mit Formelnummern gesetzt werden können.

$$c = \sqrt{a^2 + b^2} \quad (1)$$

Die Abbildung in der `multicols` Umgebung ist weiterhin ein Gleitobjekt. Obwohl Sie vom Code her auf der ersten Seite steht, wird sie auf die zweite Seite verschoben.

An dieser Stelle noch einmal der Hinweis, dass die Beispiele im Quelltext auf einem Webserver liegen, man muss sie also nicht abtippen.

```

62 \begin{figure*}[tbp]
63 \begin{verbatim}
64 \begin{figure*}[tbp]
65 \begin{verbatim}
66 ...
67 \end{verb...}
68
69 \caption{Abbildungen in der
70 \texttt{multicols}-Umgebung}
71 \end{figure*}
72 \end{verbatim}
73
74 \centering
75 \caption{Abbildungen in der
76 \texttt{multicols}-Umgebung}\label{fig}
77 \end{figure*}
78
79 In dieser Umgebung k"onnen mathematische Formeln in der
80 "ublichen Weise verwendet werden, wie die folgende
81 abgesetzte Formel zeigt:
82 \[
83 c^2=a^2+b^2a
84 \]
85 Die Umgebung \texttt{eqnarray} mit den
86 Formeln~(\ref{eqnb})--(\ref{eqne})
87 zeigt, dass sogar mehrzeilige Formeln mit
88 Formelnummern gesetzt werden k"onnen.
89 %
90 \begin{eqnarray}
91 c \&= \& \sqrt{a^2+b^2}\label{eqnb}\\
92 x \&= \& \sin t \\
93 y \&= \& \cos t \\
94 x^2+y^2 \&= \& 1 \nonumber\\
95 z \&= \& t\label{eqne}
96 \end{eqnarray}
97
98 Dass Listenumgebungen verwendet werden k"onnen, zeigt die
99 folgende \texttt{enumerate}-Liste:
100 %
101 \begin{enumerate}
102 \item Dies ist der erste Eintrag in einer
103 \texttt{enumerate}-Liste.

```

```

104 \item Es folgen weitere Einträge.
105 \item Dies ist der letzte Listeneintrag.
106 \end{enumerate}
107 %
108 Auf weitere Beispiele von Standard-Umgebungen wird hier
109 verzichtet.
110
111 \subsection*{\texttt{multicols} geschachtelt}
112 %
113 Innerhalb von \texttt{multicols} werden Überschriften in
114 der Spalte gesetzt, wie die vorstehende Überschrift zeigt.
115
116 \begin{multicols}{2}[
117   Mit diesen Zeilen über alle Spalten einer inneren
118   \texttt{multicols}-Umgebung beginnt eine innere
119   \texttt{multicols}-Umgebung.
120 ]
121 \raggedright
122 Um Probleme mit dem Randausgleich wegen der geringen
123 Zeilenlänge zu vermeiden, wird der innere Mehrspaltensatz
124 mit einem Flatterrand rechts gesetzt.
125
126 Die innere \texttt{multicols}-Umgebung ist auf die aktuelle
127 Spalte beschränkt und kann nicht umbrochen werden.
128 Ist sie länger, ragt sie über die Spalte der äußeren
129 \texttt{multicols}-Umgebung nach unten hinaus.
130 \end{multicols}
131
132 Dieser Satz steht hinter der inneren
133 \texttt{multicols}-Umgebung.
134 \end{multicols}
135 \end{document}

```

```

\begin{figure*}[tbp]
\begin{verbatim}
...
\end{verb...}

\caption{Abbildungen in der
\texttt{multicols}-Umgebung}
\end{figure*}

```

Abbildung 1: Abbildungen in der `multicols`-Umgebung

$$x = \sin t \quad (2)$$

$$y = \cos t \quad (3)$$

$$x^2 + y^2 = 1$$

$$z = t \quad (4)$$

Dass Listenumgebungen verwendet werden können, zeigt die folgende `enumerate`-Liste:

1. Dies ist der erste Eintrag in einer `enumerate`-Liste.
2. Es folgen weitere Einträge.
3. Dies ist der letzte Listeneintrag.

Auf weitere Beispiele von Standard-Umgebungen wird hier verzichtet.

`multicols` geschachtelt

Innerhalb von `multicols` werden Überschriften in der Spalte gesetzt, wie die vorstehende Überschrift zeigt.

Mit diesen Zeilen über alle Spalten einer inneren `multicols`-Umgebung beginnt eine innere `multicols`-Umgebung.

Um Probleme mit dem Randausgleich wegen der geringen Zeilenlänge zu vermeiden, wird der innere Mehrspaltensatz mit einem Flatterrand rechts gesetzt. Die innere

`multicols`-Umgebung ist auf die aktuelle Spalte beschränkt und kann nicht umbrochen werden. Ist sie länger, ragt sie über die Spalte der äußeren `multicols`-Umgebung nach unten hinaus.

Dieser Satz steht hinter der inneren `multicols`-Umgebung.

Das Paket stellt auch einige Befehle zur Steuerung des Verhaltens der `multicols`-Umgebung zur Verfügung. Einzige Ausnahme ist

```
\columnbreak
```

zum Beenden der aktuellen Spalte, sobald die aktuelle Zeile abgeschlossen ist. Der Befehl hat auf die aktuelle Spalte die gleiche Wirkung wie ein `\pagebreak`-Befehl (der auch in einer `multicols`-Umgebung verwendet werden kann) für die aktuelle Seite. Falls der Spaltenausgleich aktiviert ist, wird er auch bei der explizit beendeten Spalte durch Einfügen von zusätzlichen Abständen durchgeführt.

Zwei weitere Befehle ermöglichen das Aktivieren und Deaktivieren des Spaltenausgleichs:

```
\flushcolumns  
\raggedcolumns
```

Der erste der beiden Befehle aktiviert, der zweite deaktiviert den Spaltenausgleich. Der letzte beim Formatieren einer Seite sichtbare Befehl legt das Verhalten fest, auch wenn die Stelle, an der er steht, erst auf der folgenden Seite gesetzt wird. Wird eine Seite explizit mit einem (unbedingten) `\pagebreak`-Befehl beendet, hat selbst ein direkt folgender Befehl zur Änderung des Spaltenausgleichs keine Rückwirkung auf die gerade abgeschlossene Seite.

Im mehrspaltigen Satz sind die Zeilen ziemlich kurz und enthalten daher auch nur wenig Wortzwischenräume, die $\text{\LaTeX}$ für den Randausgleich nutzen kann. Mehr Spalten verschärfen offensichtlich das Problem. Bei sehr kurzen Zeilen ist es $\text{\LaTeX}$ häufig unmöglich, den Text in der gewohnten Qualität zu setzen, und die Folge sind überstehende und zu leere Zeilen. Das Problem lässt sich vermindern und gelegentlich sogar lösen, indem $\text{\LaTeX}$ erlaubt wird, etwas zusätzlichen Leerraum (engl. *white space*) in den Zeilen zu akzeptieren, indem der Länge `\emergencystretch` der maximal erwünschte Leerraum pro Zeile zugewiesen wird. Bei einspaltigem Satz sind häufig wenige Millimeter ausreichend. Bei zusätzlichen Spalten und damit kürzeren Zeilen wird man den Wert erhöhen müssen. `multicol` nutzt diese Möglichkeit, um mit dem Befehl

```
\setemergencystretch{<Spzahl>}{<Breite>}
```

die Länge `\emergencystretch` nach der Formel

$$\text{\code{\emergencystretch}} = \text{\code{<Spzahl>}} \times 4\text{ pt}$$

abhängig von der Anzahl der Spalten zu setzen (Stand: Version 1.6 e vom 14.2.2004). Das zweite Argument *<Breite>*, die Spaltenbreite beim Aufruf von `multicols`, wird in der vom Paket verwendeten Formel nicht berücksichtigt. Gegebenenfalls kann der Anwender die Berechnungsformel durch Redefinition des Befehls `\setemergencystretch` ändern. Selbstverständlich kann die Länge `\emergencystretch` innerhalb der Umgebung auch direkt gesetzt werden.

Die meisten Anpassungen der `multicols`-Umgebung werden durch Verändern von Längen vorgenommen. Dadurch lassen sich beispielsweise die Abstände vor und nach der Umgebung oder der Spaltenabstand ändern. Durch Vorgabe einer positiven Strichstärke ist es auch möglich, die Spalten durch vertikale Linien zu trennen. Die folgenden Längen beeinflussen das Verhalten der `multicols`-Umgebung:

`\premulticols`

Legt fest, wieviel Platz auf der Seite noch frei sein muss, wenn der Mehrspaltensatz noch auf dieser Seite beginnen soll. Andernfalls beginnt der Mehrspaltensatz auf einer neuen Seite.
(Voreinstellung: 50 pt)

`\postmulticols`

Legt fest, wieviel Platz nach Beendigung von `multicols` auf der Seite noch frei sein muss, um nachfolgenden Text noch auf dieser Seite zu setzen. Andernfalls beginnt der folgende Text auf einer neuen Seite.
(Voreinstellung: 20 pt)

`\multicolsep`

Abstand der `multicols`-Umgebung vom vorhergehenden und nachfolgenden Text (wenn er auf der gleichen Seite wie `multicols` gesetzt wird).
(Voreinstellung: 12 pt plus 4 pt minus 3 pt)

`\columnsep`

Abstand zwischen den Spalten.
(Voreinstellung: 10 pt)

`\columnseprule`

Strichstärke einer vertikalen Linie zwischen den Spalten. Ist der Wert positiv, wird die Linie gezeichnet, bei einer Strichstärke von 0 pt ist sie nicht sichtbar.
(Voreinstellung: 0 pt)

`\multicolbaselineskip`

Korrektur des Zeilenabstands innerhalb der `multicols`-Umgebung. Bei engen Spalten kann es erforderlich sein, den Zeilenabstand zu

korrigieren, mit diese Länge legt man den positiven oder negativen Korrekturabstand fest.

Sie ist speziell für Designer von Paketen gedacht und sollte nur mit Vorsicht geändert werden.

(Voreinstellung: 0 pt)

\multicolovershoot

Legt fest, dass Spalten ihre vorgegebene Länge bis zum angegebenen Wert *überschreiten* dürfen, ohne dass eine Warnung ausgegeben wird.

(Voreinstellung: 0 pt)

\multicolundershoot

Legt fest, dass Spalten ihre vorgegebene Länge bis zum angegebenen Wert *unterschreiten* dürfen, ohne dass eine Warnung ausgegeben wird.

(Voreinstellung: 0 pt)

Auch einige Zähler beeinflussen das Verhalten der `multicols`-Umgebung. Durch Setzen von Zählern lässt sich beispielsweise steuern, wie weit $\text{\LaTeX}$ über das Zeilenende hinauslesen soll, um eine geeignete Stelle für den Seitenumbruch zu finden. Zulässige Abweichungen der letzten Spalte von der Länge der übrigen Spalten lassen sich ebenfalls durch einen Zähler festlegen. Die folgenden Zähler steuern das Verhalten von `multicols`:

collectmore

Legt fest, wieviel Text der nächsten Seite berücksichtigt werden soll, bevor über den Seitenumbruch endgültig entschieden wird. Ein negativer Zählerwert führt dazu, dass ein Seitenumbruch erfolgt, sobald alle Spalten auf der Seite gefüllt sind. In diesem Fall wird der Umbruch in der Regel nicht optimal sein. Durch Vorausschau (ab dem Wert 0) lassen sich allerdings häufig bessere Stellen für den Umbruch finden, wobei höhere Werte eine bessere Optimierung ermöglichen.

Ein Problem beim Mehrspaltensatz mit Vorausschau (voreingestellt: 4 Zeilen) besteht darin, dass am Ende der Seite Material übrig bleibt, das nicht mehr auf der Seite gesetzt werden kann. Dadurch kann es passieren, dass eine Fußnote gesetzt wird, obwohl die Fußnotenmarkierung erst auf der nächsten Seite erscheint. Durch einen expliziten `\pagebreak`-Befehl lässt sich dieses Problem vermeiden. Eine andere Lösung sind negative Werte des Zählers `collectmore` (meist reicht ein Wert von -1), um beim Seitenumbruch weniger weit vorzuschauen und dadurch verursachte

Probleme zu lösen. Werte kleiner als -3 führen zu einer Verkürzung der letzten Spalte.

(Voreinstellung: 0)

unbalance

Steuert den Spaltenausgleich auf der letzten Seite und legt fest, dass die Spalten mit Ausnahme der ganz rechten Spalte um die dem Zählerwert entsprechende Anzahl von Zeilenabständen (`\baselineskip`) verlängert werden.

Im Gegensatz zu den anderen Zählern ist `unbalance` lokal für eine `multicols`-Umgebung und wird beim Verlassen der Umgebung zurückgesetzt.

(Voreinstellung: 0)

columnbadness

Maximaler Wert für die Qualität (engl. *badness*) einer Spalte. Bei einem größeren Wert wird ein Umbruch einer Seite mit der betroffenen Spalte abgelehnt. Ein Badness-Wert von 0 wird von $\text{\TeX}$ als optimal, einer von 10 000 als unendlich schlecht betrachtet. Der spezielle Wert 100 000 steht für eine Spalte, die zuviel Text enthält (`Overfull \vbox`).

Lässt sich eine den Anforderungen entsprechende Lösung nicht finden, wird der gesamte Text in die erste Spalte gesetzt.

Voreinstellung: 10 000)

finalcolumnbadness

Maximaler Wert für die Qualität der *letzten* Spalte, bis zu dem die Spalte mit Leerraum auf die Länge der anderen aufgefüllt wird. Für größere Werte wird die letzte Spalte nicht ausgeglichen.

Voreinstellung: 9 999)

tracingmulticols

Legt den Umfang der Informationen der `multicols`-Umgebung ab dem Zeitpunkt der Änderung fest. Damit können global oder lokal bei Problemen zusätzliche Informationen abgerufen bzw. unterdrückt werden, während die entsprechenden Paketoptionen das globale Verhalten festlegen. Die folgenden Werte sind vorgesehen:

- 0 entspricht `errorshow`,
- 1 entspricht `infoshow`,
- 2 entspricht `balancingshow`,
- 3 entspricht `markshow`,
- ≥ 4 entspricht `debugshow` (= 5).

(Voreinstellung: 0)

Da beim Mehrspaltensatz die Zeilen sehr kurz sind, ist eine Anpassung von `\tolerance` und `\pretolerance` erforderlich, um zusätzlichen Leerraum in den Zeilen zuzulassen. Bei den Standardwerten von $\text{\LaTeX}$ (beispielsweise 200 für `\tolerance`) sind wegen der verkürzten Zeilen die üblichen Anforderungen häufig nicht erfüllbar. Speziell für Designer von Paketen sind zwei interne Zähler vorgesehen:

```
\multicoltolerance
\multicolpretolerance
```

Diese Zähler enthalten Werte für `\tolerance` und `\pretolerance` während des Mehrspaltensatzes. Die Voreinstellungen sind 9999 für `\multicoltolerance` und -1 für `\multicolpretolerance`.

9.15 Paket rawfonts

Dieses Paket von Alan Jeffrey ist nur für ältere Dokumente von Bedeutung, die mit $\text{\LaTeX}$ 2.09 entwickelt wurden und ‚low-level‘ Fontbefehle wie `\tenrm` verwenden. Ohne Angabe von Optionen lädt das Paket mehr als 60 derartige Fonts.

Werden nur wenige Fonts benötigt, wird der $\text{\LaTeX}$ zur Verfügung stehende Speicher durch die vielen Fonts unnötig belastet. Da `rawfonts` das Paket `somedefs` anzieht, kann man in der dort vorgesehenen Form gegebenenfalls angeben, welche Fonts geladen werden sollen. Damit hat der `\usepackage`-Befehl für das Paket `rawfonts` die folgende Form:

```
\usepackage[only,<Font>[,<Font>]...]{rawfonts}
```

Die Option `only` legt fest, dass nur die nachfolgend als Optionen aufgeführten Fonts zu laden sind, beispielsweise `\tensf` für den Fontbefehl `\tensf`.

9.16 Paket showkeys

Umfangreichere Dokumente enthalten häufig Querverweise und Verweise auf die Literatur. Die Kontrolle der Verweise ist nicht immer ganz einfach. Dieses Paket von David Carlisle unterstützt diese Kontrolle, indem es die Befehle `\label`, `\ref`, `\pageref`, `\cite` und `\bibitem` so redefiniert, dass die vergebenen bzw. verwendeten Marken im gesetzten

Dokument angezeigt werden. Wahlweise lässt sich die Ausgabe der in `\cite`- und `\ref`-Befehlen verwendeten Marken unterdrücken. Eigene Befehle stellt das Paket nicht zur Verfügung.

Das Paket ist kompatibel mit der Option `fleqn` der Standardklassen. Die Verweis-Befehle und ihre Varianten in den Paketen der $\mathcal{A}\mathcal{M}\mathcal{S}$ und `varioref`, `natbib` und `harvard` werden ebenfalls unterstützt.

Das Verhalten lässt sich durch Angabe von Optionen beim Laden des Pakets steuern. Das Paket wird wie immer mit dem Befehl

```
\usepackage[<Option(en)>]{showkeys}
```

geladen. Die folgenden Optionen werden vom Paket berücksichtigt:

- notref** Unterdrückt die Ausgabe der Marken bei `\ref`- und `\pageref`-Befehlen und deren Varianten im Paket `varioref`.
- notcite** Unterdrücken der Ausgabe der Zitiermarken beim `\cite`-Befehl und dessen Varianten in den Paketen `harvard` und `varioref`.
- color** Ausgabe der Marken in anderen Farben. Voreingestellt ist ein helles Grau. Bei Angabe dieser Option das Paket `color` automatisch geladen. Das Paket unterscheidet die beiden Farben

refkey

Farbe von Marken, die referiert werden. Diese Farbe wird von allen `\ref`- und `\cite`-Befehlen und ihren Varianten verwendet.

labelkey

Farbe von Marken bei ihrer Vereinbarung. Diese Farbe wird von `\label`- und `\bibitem`-Befehlen verwendet.

Die voreingestellten Farben sind mit

```
\definecolor{refkey}{gray}{.75}
\definecolor{labelkey}{gray}{.75}
```

vereinbart. Ist eine Unterscheidung oder auch nur eine andere Farbgebung gewünscht, können die Farben redefiniert werden.

- final** Deaktiviert die Ausgabe der Marken, wenn sie nicht länger von Interesse sind, beispielsweise im fertigen Dokument.
- draft** Ist das Gegenteil der Option `final` und nur aus Gründen der Kompatibilität mit anderen Paketen vorhanden. Laut Dokumentation hat sie keine Wirkung.

Bei ihrer Vereinbarung werden die Marken in einer `\fbox` in der Regel in den Seitenrand geschrieben. In einer `minipage` ist das nicht möglich, und sie werden daher neben die `minipage` gesetzt. In einer abgesetzten nicht nummerierten Gleichung (nur im Zusammenhang mit `\pageref` sinnvoll) wird die Marke an die Gleichung angehängt. Auch ein `\label`-Befehl in einer `\caption` führt dazu, dass die Marke der `\caption` direkt folgt.

Beim Referenzieren einer Marke wird dieser in einer kleineren Schrift über dem Verweis angegeben. Die Zeilenabstände werden dabei nicht verändert.

Das Layout eines Dokuments wird durch die Ausgabe der Marken – abgesehen von den zusätzlichen Informationen – nicht verändert und stimmt mit der Ausgabe ohne Verwendung des Pakets überein. Da es keinen Einfluss auf das fertige Dokument hat und vor allem eine Hilfe bei der Arbeit mit (Literatur-)Verweisen ist, kann hier auf Beispiele verzichtet werden. Als Beispiel für die Verwendung kann auch die Originaldokumentation dienen. Darüberhinaus empfiehlt es sich, bei Bedarf das Paket zu laden, und sich das Ergebnis anzusehen.

9.17 Paket `somedefs`

Unterstützung bei der Entwicklung eigener Pakete leistet dieses Paket von Alan Jeffrey. Für den reinen Anwender ist es hingegen nicht gedacht. Er kommt mit ihm nur dann in Berührung, wenn ein von ihm verwendetes Paket `somefonts` verwendet. Typischerweise wird das Paket daher nicht mit `\usepackage` geladen, sondern in einem anderen Paket mit `\RequirePackage{somedefs}` angezogen. Ein Beispiel für die Verwendung ist das vom gleichen Autor entwickelte Paket `rawfonts`.

Beim Einbinden von `somedefs`, kann ausgewählt werden, ob alle oder nur ausgewählte Definitionen des Pakets aktiviert werden.

Für Definitionen, die aktiviert werden sollen, ist der Befehl

```
\ProvidesDefinition{<Definition>}
```

vorgesehen. `<Definition>` besteht dabei aus einem `\newcommand`-Befehl, der den zu definierenden Befehl `\<Befehlsname>` definiert. Statt `\newcommand` können gegebenenfalls auch `\renewcommand` oder `\def` verwendet werden.

`<Definition>` wird allerdings nur wirksam, wenn sie aktiviert wird. Die Aktivierung erfolgt entweder durch den expliziten Befehl

```
\UseDefinition{\<Befehlsname>}
```

oder mit

```
\UseAllDefinitions
```

zur Aktivierung aller Befehle.

Möchte man nur einzelne Befehle Aktivieren oder die globalen Aktivierung wieder rückgängig machen, so verwendet man

```
\UseSomeDefinitions
```

Mindestens einer der Befehle `\UseAllDefinitions` bzw. `\UseSomeDefinitions` sollte verwendet werden.

Als Beispiel stellt die Dokumentation einen Ausschnitt aus einem fiktiven Paket `fred` vor, das zwei Befehle `\foo` und `\bar` definiert, die aktiviert werden können. Der Ausschnitt mit den Zeilen, die sich auf das Paket `somedefs` beziehen, lautet:

```
\RequirePackage{somedefs}
\UseAllDefinitions
\DeclareOption{only}{\UseSomeDefinitions}
\DeclareOption*{\UseDefinition{\CurrentOption}}
\ProcessOptions
\ProvidesDefinition{\newcommand{foo}}{...}
\ProvidesDefinition{\newcommand{bar}}{...}
```

Benötigt der Befehl `\bar` auch den Befehl `\foo`, sollte `bar` explizit als Option durch

```
\DeclareOption{bar}{\UseDefinition{bar}%
\UseDefinition{foo}}
```

vereinbart werden. Beim Laden des Pakets mit

```
\usepackage[only,foo]{fred}
```

setzt der Befehl `\UseSomeDefinitions` der durch die Option `only` aktiviert wird, den Befehl `\UseAllDefinitions` – die Voreinstellung – außer Kraft.

9.18 Paket tabularx

Auch diese Erweiterung der `tabular`-Umgebung stammt von David Carlisle. Das Paket stellt eine Umgebung `tabularx` zur Verfügung, die ähnlich zur Umgebung `tabular*` die Vorgabe der Tabellenbreite möglich macht.

Der Unterschied zwischen den beiden Umgebungen besteht in den Anpassungen, die zum Erzielen der vorgegebenen Tabellenbreite vorgenommen werden. Bei `tabular*` werden dazu diejenigen Spaltenabstände verändert, die in der Tabellendefinition als `\extracolsep{\fill}` angegeben sind. `tabularx` variiert im Gegensatz dazu die Breite spezieller Spalten, die Text mit Zeilenumbrüchen setzen.

Die Form der Umgebung stimmt überein mit der `tabular*`-Umgebung:

```
\begin{tabularx}{<Breite>}[<Pos>]{<Spvereinbarung>}
  \item...
\end{tabularx}
```

Zusätzlich zu den üblichen Spaltentypen kommt der Spaltentyp `x` hinzu, der einer Spaltenangabe `p{...}` entspricht. Die Breite wird dabei von $\text{\LaTeX}$ geeignet gewählt, so dass die Tabelle insgesamt die vorgegebene Breite hat. Der zum Erreichen der vorgegebenen Tabellenbreite erforderliche Platz wird auf alle `x`-Spalten gleichmäßig verteilt. Die Breite einer `x`-Spalte kann allerdings nicht negativ werden, sondern sie beträgt mindestens `1em`, auch wenn die Tabelle dadurch breiter wird als vorgegeben. Da das Paket `tabularx` das `array` verwendet, können alle Erweiterungen von `array` genutzt werden. Insbesondere steht auch der `\newcolumnntype`-Befehl zur Verfügung.

Die beiden folgenden Beispiele zeigen eine Anwendung der `tabularx`-Umgebung. Sie unterscheiden sich lediglich in der vorgegebenen Breite der Tabelle.

```
11 \begin{tabularx}{10cm}[b]{|>{\bfseries}c|X|X|c|}
12 \hline
13 \multicolumn{2}{|c|}{Zweispaltiger Eintrag}
14 & Langer Eintrag & ABC \\\
15 \hline
16 aaa & Ein langer Eintrag "uber mehrere
17 Zeilen & Zweiter langer Eintrag & bbb \\\
18 \hline
19 AAA & Text & Text & BBB \\\
20 \hline
21 \end{tabularx}
22 \rule{3mm}{1.4pt}
```

```
23 \end{center}
24 \end{document}
```

Zweispaltiger Eintrag		Langer Eintrag	ABC
aaa	Ein langer Eintrag über mehrere Zeilen	Zweiter langer Eintrag	bbb
AAA	Text	Text	BBB

[Buch1/Ergaenz/tabularxa.tex]

```
11 \begin{tabularx}{7cm}[b]{|>\bfseries c|X|X|c|}
12 \hline
13 \multicolumn{2}{|c|}{Zweispaltiger Eintrag}
14 & Langer Eintrag & ABC \\
15 \hline
16 aaa & Ein langer Eintrag "uber mehrere
17 Zeilen & Zweiter langer Eintrag & bbb \\
18 \hline
19 AAA & Text & Text & BBB \\
20 \hline
21 \end{tabularx}
22 \rule{3mm}{1.4pt}
23 \end{center}
24 \end{document}
```

Zweispaltiger Eintrag		Langer Eintrag	ABC
aaa	Ein langer Eintrag über mehrere Zeilen	Zweiter langer Eintrag	bbb
AAA	Text	Text	BBB

[Buch1/Ergaenz/tabularxb.tex]

Ein Unterschied zwischen den Umgebungen `tabular` und `tabular*` auf der einen Seite und `tabularx` auf der anderen besteht auch darin, dass `tabular` und `tabular*` ohne Einschränkungen ineinander geschachtelt werden können, während eine innere `tabularx`-Umgebung in einem `tabularx` in geschweifte Klammern `{ }` eingeschlossen werden *muss*. In `tabularx` dürfen darüber hinaus auch keine zerbrechlichen Befehle wie `\verb` verwendet werden.

Das Verhalten der `tabularx`-Umgebung kann in einigen Punkten angepasst werden. Dazu stellt das Paket einige Befehle zur Verfügung. Informationen über die gefundenen Spaltenbreiten und die Versuche zu ihrer Bestimmung können durch Angabe von Paketoptionen (beim Laden des Pakets oder global im `\documentclass`-Befehl) abgerufen werden. Sie werden am Bildschirm angezeigt und in die `.log`-Datei geschrieben. Die beiden folgenden Optionen des Pakets haben die gleiche Bedeutung:

debugshow | infoshow

Geben als Kontrollinformation die getesteten Spaltenbreite für die `x`-Spalten und die endgültige Spaltenbreite für alle Umgebungen `tabularx` aus.

Statt der Angabe einer dieser beiden Optionen kann in der Präambel auch der Befehl

```
\tracingtabularx
```

angegeben werden. Er bewirkt, dass für alle `tabularx`-Umgebungen, die danach gesetzt werden, die Kontrollinformationen ausgegeben werden. Er kann auch nach `\begin{document}` verwendet werden.

In Spalten mit Zeilenumbruch sind über `>\{...\}` auch Angaben wie `\centering`, `\raggedright` oder `\raggedleft` möglich, um vom Blocksatz abweichende Ausgaben zu erzielen. Solange ein mehrzeiliges Feld mit `&` abgeschlossen wird, treten dabei keine Probleme auf. Bei der letzten – mit `\\` abgeschlossenen – Spalte einer Zeile tritt aber ein Fehler beim Formatieren auf, der dadurch verursacht wird, dass die erwähnten Befehle `\\` redefinieren. Um diesen Fehler zu umgehen stellt das Paket `tabularx` den Befehl

```
\arraybackslash
```

zur Verfügung. Mit Hilfe dieses Befehls kann auch das letzte Feld einer Tabellenzeile mehrzeilig sein ohne im Blocksatz gesetzt zu werden. Das folgende Beispiel führt zu einem Fehler beim Formatieren, wenn der Befehl `\arraybackslash` in Zeile 11 gelöscht wird.

```
10 \begin{tabularx}{10cm}[b]{|>\bfseries c|>\centering X|}%
11 >\raggedleft\arraybackslash X|}
```

Zweispaltiger Eintrag		Langer Eintrag
aaa	Ein langer Eintrag über mehrere Zeilen	Zweiter langer Eintrag
AAA	Text	Text

[Buch1/Ergaenz/tabularxabs.tex]

Das gleiche Problem tritt übrigens auch in einer `tabularx`-Umgebung auf, wenn das Paket `tabularx` geladen ist. Es lässt es sich für die Spaltentypen `p`, `m` und `b` in der gleichen Weise wie im Beispiel für die `x`-Spalte lösen.

Vordefiniert im Paket `tabularx` ist, dass alle Spalten `x` als `p{...}` mit einer passenden Breite gesetzt werden. Der Spaltentyp `x` wird dazu im Befehl

```
\tabularxcolumn
```

wie folgt definiert:

```
\newcommand{\tabularxcolumn}[1]{p#1}
```

Das Argument `#1` ist dabei die vom Paket `tabularx` eingesetzte Spaltenbreite. Diese Definition kann mit Hilfe des `\renewcommand`-Befehls verändert werden. Der vordefinierte Spaltentyp `p` kann in `m` oder `b` abgeändert werden. Ergänzend können Angaben zu Spaltentrennungen sowie `>{...}` und `<{...}` hinzugefügt werden. Der Spaltentyp `x` könnte etwa durch den folgenden Befehl redefiniert werden:

```
\renewcommand{\tabularxcolumn}%
[1]{>{\centering%
\arraybackslash}b#1}
```

Nach dieser Redefinition werden die umbrochenen Zeilen der `x`-Spalten zentriert gesetzt, und die unterste dieser Zeilen steht auf der Grundlinie.

Ein Nachteil bei den `tabularx`-Umgebungen scheint zu sein, dass alle Spalten `x` gleich breit sind. Es ist aber durchaus möglich, den verfügbaren Platz durch Ändern der Länge `\hsize` auf die einzelnen Spalten unterschiedlich zu verteilen. `\hsize` enthält nämlich die berechnete Spaltenbreite. Für eine lokale Anpassung der Spaltenbreite einer `x`-Spalte kann die Angabe

```
>{\hsize=<Faktor>\hsize}X
```

verwendet werden. Dabei ist darauf zu achten, dass die Summe der Faktoren in der Tabellendefinition identisch zur Anzahl der in der Breite modifizierten X-Spalten ist, da sich andernfalls die Tabellenbreite ändert. Außerdem sollte sich kein `\multicolumn`-Eintrag über eine X-Spalte erstrecken, da ein zu langer `\multicolumn`-Eintrag zu einer fehlerhaften Formatierung der Tabelle führt.

```
10 \begin{tabularx}{10cm}[b]{|>{\hspace=.7\hspace%
11 \centering}X|c|X|>{\hspace=1.3\hspace}X|}
```

Mehrzeiliger Eintrag	ab	Eintrag über mehrere Zeilen	Dritter langer Eintrag
AAA	1	Text	Text

[Buch1/Ergaenz/tabularxd.tex]

Die Breite der ersten Spalte in diesem Beispiel beträgt 70 % der Standardbreite. Die vierte Spalte ist 1,3 mal so breit wie die dritte Spalte, die die vom Paket berechnete Breite hat.

9.19 Paket theorem

Dieses Paket von Frank Mittelbach erweitert die Möglichkeiten der `theorem`-Umgebung und stellt unterschiedliche Stile für diese Umgebung zur Verfügung.

Zunächst wurden die Pakete `theorem` und `amsthm` (S. 433) für Mathematiker entwickelt, deren Publikationen etwas vereinfacht ausgedrückt nur aus Sätzen, Hilfssätzen, Definitionen, Beweisen, Korrolaren, Lemmas, Theoremen und ähnliche Strukturen bestehen. Man kann die Theorem-Umgebungen durchaus für eigene Strukturierungen in „normalen“ Dokumenten unterhalb der üblichen Überschriftengliederung einsetzen, um ein Dokument übersichtlicher und gleichzeitig einheitlich zu gestalten. Welcher Leser würde es nicht schätzen, beispielsweise ‚Tipps‘ oder ‚Hinweise‘ in einer abgesetzten Struktur zu finden, statt im Fließtext versteckt?

Nach dem Laden des Pakets mit

```
\usepackage{theorem}
```

sind Änderungen an einem vorhandenen Dokument in der Regel nicht erforderlich, da der Befehl `\newtheorem` sich wie bisher verhält. Eine Einschränkung ist allerdings, dass der `\newtheorem`-Befehl nach dem Laden des Pakets nur noch in der Präambel verwendet werden darf. Gegebenenfalls müssen eigene `\newtheorem`-Befehle also verschoben werden. Ohne weitere Anpassungen ändert sich die Ausgabe allerdings nicht.

```
\newtheorem{<UName>}{<Bezeichnung>}[<Zähler>]
```

Der `\newtheorem`-Befehl ist – abgesehen davon, dass er nur vor `\begin{document}` verwendet werden darf – unverändert. Auf eine Beschreibung kann daher an dieser Stelle verzichtet werden.

Die Vorteile des Pakets zeigen sich erst, wenn die in ihm zur Verfügung gestellten Befehle zur Anpassung der ‚Theorem‘-Umgebungen eingesetzt werden. Eine zentrale Rolle spielt dabei die Auswahl eines Stils für einzelne oder alle ‚Theorem‘-Umgebungen. Der Befehl zur Auswahl eines ‚Theorem‘-Stils hat die Form

```
\theoremstyle{Stil}
```

und kann ebenfalls *nur in der Präambel* verwendet werden. Er führt dazu, dass alle anschließend mit `\newtheorem` vereinbarten ‚Theorem‘-Umgebungen in diesem Stil gesetzt werden.

Die einzelnen Stile sind in Form von eigenständigen Paketen implementiert und können daher auch direkt geladen werden. Da dabei der geladene Stil nicht automatisch aktiviert wird, empfiehlt sich dieses Vorgehen allerdings nicht. Vorgesehen sind die folgenden Stile:

plain	Entspricht dem Standardstil von $\text{\LaTeX}$ und weicht allenfalls in den Abständen vor und nach der Umgebung ab. Der Inhalt der Umgebung wird kursiv (<code>\itshape</code>) gesetzt. (Paketname: thp)
change	Vertauscht im Titel Nummer und Text; Inhalt in <code>\slshape</code> . (Paketname: thc)
margin	Setzt die Nummer der <code>theorem</code> -Umgebung in den Seitenrand; Inhalt in <code>\slshape</code> . (Paketname: thm)

break	Entspricht <code>plain</code> , beginnt aber nach dem Titel eine neue Zeile; Inhalt in <code>\slshape</code> . (Paketname: thb)
changebreak	Entspricht <code>change</code> , beginnt aber nach dem Titel eine neue Zeile; Inhalt in <code>\slshape</code> . (Paketname: thcb)
marginbreak	Entspricht <code>margin</code> , beginnt aber nach dem Titel eine neue Zeile; Inhalt in <code>\slshape</code> . (Paketname: thmb)

Abgesehen vom Theoremstil `plain` wird der Inhalt der Umgebungen immer in geneigter Schrift (`\slshape`) gesetzt. Da `\newtheorem` seine Einstellungen stets global vornimmt, kann die Wirkung eines `\theoremstyle`-Befehls wie immer durch einen Block (Einschluss in geschweifte Klammern) auf die `\newtheorem`-Befehle in dieser Gruppe beschränkt werden.

Zwei weitere Befehle dienen dazu, die Fonts für Titel und Inhalt festzulegen. Zunächst der Befehl

```
\theorembodyfont{Fontbefehle}
```

der die Schriftfamilie für den Inhalt der anschließend mit `\newtheorem` vereinbarten ‚Theorem‘-Umgebungen festlegt. Bei fehlender Angabe oder einem leeren Argument wird die im Theoremstil festgelegte Schriftart verwendet. `<Fontbefehle>` steht für einen oder mehrere Befehle, die ab der aktuellen Stelle die Schriftart umsetzen, beispielsweise `\sffamily`, `\upshape` oder `\bfseries`. Auch die Wirkung dieses Befehls kann durch einen Block beschränkt werden. Die Schriftart des Inhalts verschiedener ‚Theorem‘-Umgebungen kann also unterschiedlich gewählt werden.

Der zweite Befehl legt den Font für die Titel aller ‚Theorem‘-Umgebungen fest:

```
\theoremheaderfont{Fontbefehle}
```

Für das Argument `<Fontbefehle>` gilt das oben gesagte. Da diese Festlegung für alle ‚Theorem‘-Umgebungen Gültigkeit hat, sollte der Befehl nur einmal in der Präambel verwendet werden. Eine explizite Festlegung unterschiedlicher Schriftarten für den Titel verschiedener ‚Theorem‘-Umgebungen ist ohne Änderungen an den Theoremstilen selbst nicht möglich.

Aufgrund der Implementierung ‚erbt‘ der Titelfont Eigenschaften der Body-Schriftart. Nach `\theorembodyfont{\itshape}` und `\theoremheaderfont{\bfseries}` wird der Titel kursiv und fett gesetzt.

Zusätzlich verwendet das Paket für alle ‚Theorem‘-Umgebungen die beiden Längen

```
\theorempreskipamount
\theorempostskipamount
```

zur Festlegung des Abstandes einer ‚Theorem‘-Umgebung vom vorhergehenden Text (`\theorempreskipamount`) und vom nachfolgenden Text (`\theorempostskipamount`). Die beiden Längen können dehnbare Anteile (plus und minus) enthalten.

Das folgende Beispiel zeigt unter anderem die verschiedenen Theoremstile. Auch die Befehle `\theorembodyfont` und `\theoremheaderfont` werden verwendet.

```
8 \usepackage{theorem}
9 \theoremheaderfont{\bfseries}
10 \theoremstyle{plain}
11 \newtheorem{Sz}{Satz}[section]
12 \theoremstyle{change}
13 {\theorembodyfont{\ttfamily}%
14 \newtheorem{Hs}[Sz]{Hilfssatz:}}
15 {\theoremstyle{margin}\theorembodyfont{\sffamily}%
16 \newtheorem{Lm}{Lemma}}
17 \theoremstyle{break}
18 {\theorembodyfont{\sffamily}%
19 \newtheorem{Am}{Anmerkung}[section]}
20 \theoremstyle{marginbreak}
21 {\theorembodyfont{\sffamily\slshape}%
22 \newtheorem{Th}[Lm]{Theorem}}
23 \theoremstyle{changebreak}
24 {\theorembodyfont{\upshape}%
25 \newtheorem{Bm}[Am]{Bemerkung}}
26 %
27 \begin{document}
28 \section{Theorem-Umgebungen}
29 %
30 Das Dokument zeigt verschiedene Varianten der
31 Theorem-Umgebung.
32
33 \begin{Sz}
34 Ein Satz wird im Theoremstil "\plain" gesetzt.
35 \end{Sz}
36
```

```

37 \begin{Hs}
38 Dieser Hilfssatz verwendet "\change" als Stil. Er
39 wird außerdem in Schreibmaschinenschrift gesetzt.
40 \end{Hs}
41
42 \begin{Lm}
43 Lemmas, oder doch besser Lemmata, werden im Stil
44 "\margin" serifenlose Schrift (Sans Serif) ausgegeben.
45 \end{Lm}
46
47 \begin{Th}
48 Für Theoreme wurde der Stil "\marginbreak" und
49 "schrage serifenlose Schrift gewählt.
50 \end{Th}
51
52 \begin{Am}
53 Anmerkungen werden im Stil "\break" in
54 \texttt{sffamily} gesetzt.
55 \end{Am}
56
57 \begin{Bm}
58 Für Bemerkungen verwenden wir den Stil
59 "\changebreak" und aufrechte Standardschrift.
60 \end{Bm}
61
62 \section{Ein neues Kapitel}
63 %
64 \begin{Th}Theoreme und Lemmata werden
65 durchnummeriert!\end{Th}
66
67 \begin{Sz}Sätze und Hilfssätze sind
68 kapitelweise nummeriert.\end{Sz}
69
70 \begin{Bm}Und hier folgt noch eine Bemerkung
71 in einem neuen Kapitel.\end{Bm}
72
73 Und Tschüss.
74 \end{document}

```

1 Theorem-Umgebungen

Das Dokument zeigt verschiedene Varianten der Theorem-Umgebung.

Satz 1.1 Ein Satz wird im Theoremstil „plain“ gesetzt.

1.2 Hilfssatz: Dieser Hilfssatz verwendet „change“ als Stil. Er wird außerdem in Schreibmaschinenschrift gesetzt.

1 Lemma Lemmas, oder doch besser Lemmata, werden im Stil „margin“ serifenlose Schrift (Sans Serif) ausgegeben.

2 Theorem

Für Theoreme wurde der Stil „marginbreak“ und schräge serifenlose Schrift gewählt.

Anmerkung 1.1

Anmerkungen werden im Stil „break“ in sfamily gesetzt.

1.2 Bemerkung

Für Bemerkungen verwenden wir den Stil „changebreak“ und aufrechte Standardschrift.

2 Ein neues Kapitel

3 Theorem

Theoreme und Lemmata werden durchnummeriert!

Satz 2.1 Sätze und Hilfssätze sind kapitelweise nummeriert.

2.1 Bemerkung

Und hier folgt noch eine Bemerkung in einem neuen Kapitel.

Und Tschüss.

[Buch1/Ergaenz/theorem.tex]

9.20 Paket trace

Bei der Entwicklung eigener Makros ist es nicht ungewöhnlich, dass die Ergebnisse zunächst nicht den Vorstellungen entsprechen oder sogar

Fehler verursacht werden. Falls sich das Problem nicht mit Hilfe einiger Kontrollausgaben bereinigen lässt, kommt man meist nicht darum herum, den Ablauf der Formatierung genauer zu verfolgen. Der englische Begriff *emphtrace*, eine Spur aufzeichnen, ist gleichzeitig der Name des Pakets. Ein Protokoll aller Befehle erhält man durch Angabe von `\tracingall`. Dies erzeugt eine sehr umfangreiche Ausgabe, aus der die interessierenden Informationen mühsam herausgelesen werden müssen.

Das Paket *trace* vom Frank Mittelbach – dem Entwickler einer Reihe wichtiger Standardpakete – stellt Befehle zur Verfügung, mit deren Hilfe der Umfang des Protokolls eingeschränkt werden kann. Um die Ausgabe zu reduzieren, definiert das Paket eine Reihe von internen Befehlen neu, so dass bei ihnen der Trace-Modus abgeschaltet wird. Dazu muss lediglich mit

```
\usepackage[<Option>]{trace}
```

das Paket geladen werden. Vorgesehen ist lediglich eine Option:

logonly Unterdrückt die Bildschirmausgabe des Trace-Modus und gibt die Informationen nur in die `.log`-Datei aus.

Am Bildschirm ist eine Analyse der Ausgaben ohnehin kaum möglich. Den `.log` kann man hingegen in Ruhe studieren, man kann blättern und suchen. Daher ist das Verwenden der Option sicher von Vorteil.

Zum Starten und Beenden des Trace-Modus bietet das Paket zwei Befehle an:

```
\traceon
\traceoff
```

`\traceon` verhält sich wie `\tracingall`. Wird die Version ϵ -TeX im ‚extended mode‘ verwendet (bei den heutigen TeX-Installationen der Standard), werden zusätzlich `\tracingassigns` und `\tracinggroups` aktiviert. `\traceoff` beendet den Trace-Modus wieder. Blockklammern zum Beschränken des Trace-Modus auf möglichst wenige Befehle sind damit nicht erforderlich.

Als Beispiel zeigt die Dokumentation zum Paket einen Ausschnitt aus dem Trace eines `\setlength`-Befehls bei Verwendung des Pakets *calc*, der mit Hilfe von *calc* auf die Hälfte reduziert werden kann. Für eine dort ebenfalls angegebene Beispieldatei wird der `.log` sogar um 75% verkürzt.

9.21 Paket varioref

Die Behandlung von Querverweisen ist eine der Stärken von $\text{\LaTeX}$. Da die Verweise von $\text{\LaTeX}$ aufgelöst werden, muss der Autor eines Textes gar nicht wissen, wo das Ziel des Verweises liegt. Für den Leser hingegen ist es oft hilfreich, wenn ihm bei ‚weit entfernt‘ liegenden Verweisen die Seitennummer mitgeteilt wird, auf die verwiesen wird. Speziell bei Gleitobjekten kann es zu Verschiebungen über viele Seiten kommen, wenn das Objekt nicht sofort gesetzt werden kann. Wird das Gleitobjekt allerdings auf der gleichen Seite gesetzt wie der Verweis, macht es wenig Sinn, mit `\pageref` auf die (aktuelle) Seite zu verweisen.

Das Paket `varioref` von Frank Mittelbach löst die meisten dieser Probleme, indem es Befehle zur Verfügung stellt, die sich an die aktuelle Situation anpassen und unterschiedlich bei Verweisen auf die aktuelle, die vorhergehende, die folgende oder andere Seiten reagieren. Da dabei Text hinzugefügt wird, sollte beim Laden des Pakets mit

```
\usepackage[<Option(en)>]{varioref}
```

die zu verwendende Sprache unbedingt als Teil von `<Option(en)>` angegeben werden. Unterstützt werden die meisten Sprachoptionen des `babel`-Pakets:

afrikaans, american, austrian, brazil, breton, catalan, croatian, czech, danish, dutch, english, esperanto, finnish, francais, french, galician, german, germanb, greek, italian, magyar, naustrian, ngerman, norsk, nynorsk, polish, portuges, romanian, russian, slovak, slovene, spanish, swedish, turkish

Die Optionen `francais` und `germanb` sind Synonyme für `french` und `german`. Bei ihrer Verwendung wird man aufgefordert, die eigentlichen Optionen `french` oder `german` zu verwenden.

Neben den Sprachoptionen berücksichtigt `varioref` noch die beiden folgenden Optionen:

draft Fehlermeldungen werden als Warnungen behandelt.
final Ausgabe eines Fehlers, wenn innerhalb eines Verweises auf eine Seite ein Seitenumbruch erfolgt.
 (Voreinstellung)

Dieser vom Paket ausgelöste Fehler wird am Ende der Paketbeschreibung auf Seite 308 behandelt.

Statt der Optionen können in der Präambel die beiden folgenden Befehle angegeben werden:

```
\vrefwarning
```

Entspricht der Option `draft`.

```
\vrefshowerrors
```

Entspricht der Option `final`.

Eine zentrale Rolle spielen in diesem Paket die Befehle

```
\vref{<Marke>}
\vref*{<Marke>}
\fullref{<Marke>}
```

die auf eine mit dem üblichen `\label`-Befehl vereinbarten `<Marke>` verweisen.

Im Unterschied zu einem `\ref`-Befehl entfernt `\vref` zunächst jeglichen Leerraum (engl. *white space*) vor dem Befehl und fügt dann einen eigenen Abstand ein. Dadurch spielt es keine Rolle, ob `\vref` direkt an den vorhergehenden Text angehängt wird oder beispielsweise ein Leerzeichen folgt. Der eingesetzte Text entspricht in der Regel durchaus den üblichen Gepflogenheiten.

In gewissen Fällen ist allerdings Leerraum vor der eingesetzten Nummer nicht erwünscht, etwa bei Gleichungsnummern, die in runde Klammern stehen. Für solche Fälle gibt es die Variante `\vref*`, die keinen Abstand einfügt. Da weiterhin jeglicher Leerraum entfernt wird (selbst explizite Leerzeichen), wird die Nummer in jedem Fall direkt an den vorhergehenden Text angehängt.

`\fullref` ist ein vereinfachter Befehl zum Verweis auf Marken, die sicher nicht auf der aktuellen Seite oder benachbarten Seiten liegen. Dieser Befehl wird zur Verfügung gestellt, da bei jeder Verwendung des `\vref`-Befehls intern zwei Makros definiert werden, um die genaue Position des Verweises zu verfolgen und damit gegebenenfalls bei Problemen Fehler melden zu können. Bei sehr häufiger Verwendung von `\vref` in einem Dokument kann es daher vorkommen, dass $\TeX$ wegen fehlendem Speicher (‘name space’ oder ‘main memory’) die Formatierung abbrechen muss. In derartigen Fällen sollte dort, wo es möglich ist, der `\fullref`-Befehl verwendet werden. Hinsichtlich des Leerraums verhält sich `\fullref` wie der `\ref`-Befehl.

Bei einem Verweis auf ein Ziel, dass sich auf einer anderen Seite befindet, wird ein Verweis auf diese Seite eingefügt. Dieser Verweis besteht aus dem Text „auf Seite“ und der Seitennummer. Ist die Verweiseite in unmittelbarer Nachbarschaft, so werden entsprechende Texte ohne Seitennummer ausgegeben. Der bei einem Seitenverweis eingesetzte Text ist natürlich abhängig von der Sprache des Dokuments. Daher werden die folgenden Befehle zur Definition der Verweistexte verwendet:

`\reftextafter`

Verweis auf die folgende Seite (bei zweiseitigem Layout: nicht sichtbar);

Voreinstellung: auf der nächsten Seite

`\reftextbefore`

Verweis auf die vorhergehende Seite (bei zweiseitigem Layout: nicht sichtbar);

Voreinstellung: auf der vorherigen Seite

`\reftextfaceafter`

Bei zweiseitigem Layout Verweis auf die folgende rechte Seite;

Voreinstellung: auf der nächsten Seite

`\reftextfacebefore`

Bei zweiseitigem Layout Verweis auf die vorhergehende linke Seite;

Voreinstellung: auf der vorherigen Seite

`\reftextfaraway{<Marke>}`

Verweis auf eine nicht angrenzende Seite.

Voreinstellung: auf Seite~\pageref{#1}

Angegeben sind die Voreinstellungen für die deutsche Sprache, wobei die Texte als $\LaTeX$ -Eingabe angegeben sind (beispielsweise bedeutet ~ dabei ein Leerzeichen, an dem kein Zeilenumbruch erfolgen kann).

Die Voreinstellungen lassen sich mit `\renewcommand`-Befehlen ändern. Mit Ausnahme von `\reftextfaraway` haben die Befehle keine Argumente. `\reftextfaraway` hat genau ein Argument, und zwar die Marke, die das Ziel des Verweises ist. Eine Unterscheidung zwischen benachbarten und weiter entfernt liegenden Seiten ist übrigens nur dann möglich, wenn die Seiten ‚normal‘ (mit `\pagenumbering{arabic}`) nummeriert sind.

Zur Verwendung bei der Definition obiger Befehle stehen noch

```
\reftextvario{TextA}{TextB}
\thevpagerefnum
```

zur Verfügung. `\reftextvario` erlaubt es, Verweise abwechslungsreicher zu gestalten, indem der Befehl alternierend `<TextA>` bzw. `<TextB>` einfügt und so für Varianz bei den Verweisen sorgt.

Der Befehl `\thevpagerefnum` ermöglicht in den Befehlen `\reftextafter`, `\reftextbefore` und `\reftextface...`, die weder die Seite noch die Bezeichnung der Marke, auf die verwiesen wird kennen, trotzdem die Seitennummer zu verwenden.

Möglicherweise möchte man in einem Dokument stets mit

```
Formel (\ref{<Marke>})
```

auf Gleichungen verweisen. Dazu muss man ohne das Paket `varioref` entweder die vollständige Formulierung bei jedem Verweis wiederholen oder sich einen eigenen Befehl für den Verweis definieren. Durch

```
\labelformat{<Zähler>}{<Vtext>}
```

kann man einen solchen Text – abhängig vom Typ des Zählers, auf den verwiesen wird – festlegen. `<Zähler>` ist dabei ein Zähler wie `section`, `equation` oder `figure`, für den der bei einem Verweis zu verwendende Text `<Vtext>` festgelegt wird.

Bei der Verwendung im Text wird `<Vtext>` vom Paket mit der einzusetzenden Darstellung des Zählers als *einzigem* Argument verwendet. `<Vtext>` muss also `#1` an der Stelle enthalten, an der die Darstellung des Zählers eingesetzt werden soll. Der `\label`-Befehl verwendet diese Information: Statt lediglich die Darstellung des Zählers in die `.aux`-Datei zu schreiben, wird `<Vtext>` mit der eingesetzten Darstellung des Zählers protokolliert. Beim nächsten Formatieren wird dann der komplettierte Verweistext vom `\ref`-Befehl eingesetzt. Ein Beispiel für den `\labelformat`-Befehl für Verweise auf Gleichungen ist:

```
\labelformat{equation}{(#1)}
```

Damit werden Formelnummern automatisch in runde Klammern gesetzt. Wenn generell das Wort *Formel* hinzugefügt werden soll, könnte der Befehl auch lauten:

```
\labelformat{equation}{Formel~(#1)}
```

Beginnt nun `<Vtext>` mit einem kleingeschriebenen Wort, könnte ein Satz nicht mit einem `\ref`-Befehl beginnen. Aus diesem Grund sieht das Paket die Varianten

```
\Ref{<Marke>}
\Vref{<Marke>}
```

zur Verfügung, die den ersten Buchstaben (einen Kleinbuchstaben) von `<Vtext>` in einen Großbuchstaben umwandelt. Diese Umwandlung setzt allerdings voraus, dass das erste Zeichen von `<Vtext>` ein einfacher Buchstabe ist. Andernfalls klappt die Umwandlung nicht, und es kann sogar zu Fehlern beim Formatieren kommen. Ist tatsächlich die Umwandlung anderer Zeichen – etwa von Buchstaben mit Akzenten – erforderlich, muss der gesamte von der Umwandlung betroffene Teil in geschweifte Klammern eingeschlossen werden. Als Beispiel erwähnt die Dokumentation den Verweis auf eine Abbildung in Ungarisch in der Form

```
\labelformat{figure}{\{'a}bra~\thefigure}.
```

Verweise mit `\ref`/`\Ref`, `\vref`/`\Vref` oder `\vref*` sind nur sinnvoll, wenn dem Ziel des Verweises eine Kennung, also die Darstellung eines Zählers, zugeordnet ist. Andernfalls kann man nur auf eine Seite verweisen.

Die gerade beschriebenen Referenz-Befehle sind Kombinationen des Verweises auf eine Kennung mit zugehöriger Seite wobei die intelligente Behandlung des Verweises auf die Seite einen wesentlichen Aspekt darstellt. Wird lediglich ein Seitenverweis erzeugt, bleiben die gemachten Überlegungen dennoch gültig. Um eine entsprechend Behandlung reiner Seitenverweise zu ermöglichen, stellt das Paket analoge Befehle zur Verfügung:

```
\vpageref[<TextA>][<TextB>]{<Marke>}
\vpageref*[<TextA>][<TextB>]{<Marke>}
```

Zum Verweis auf die Seite, auf der `<Marke>` vereinbart ist, wird wie bei `\vref` unter Verwendung der definierten Texte verwiesen, falls keine optionalen Argumente angegeben sind und das Ziel des Verweises nicht auf der aktuellen Seite liegt. `\vpageref` und `\vpageref*` verhalten sich hinsichtlich Leerraum vor den Befehlen wie `\vref` und `\vref*`. Um zu verhindern, dass bei einem Verweis auf die aktuelle Seite wie bei `\vref` gar nichts gesetzt wird, verwenden `\vpageref` und `\vpageref*` in diesem Fall den zusätzlichen Befehl

```
\reftextcurrent
```

Verweis auf die aktuelle Seite.

Voreinstellung: auf dieser Seite

und setzt den ihm zugeordneten Text.

Die optionalen Argumente erlauben eine weitergehende Steuerung des auszugebenden Texts. `<TextA>` wird – falls angegeben – bei einem Verweis auf die aktuelle Seite verwendet. Ist beispielsweise bekannt, dass eine Abbildung *vor* dem Verweis gesetzt wird, könnte

```
\vpageref[oben]{fig:foo}
```

verwendet werden.

Das zweite optionale Argument `<TextB>` ist ein Text, der der eigentlichen Seitenreferenz vorangestellt wird. Damit ist es beispielsweise möglich, Wortumstellungen bei einer Referenz auf die aktuelle Seite und auf andere Seiten zu berücksichtigen. So könnte ein Verweis auf eine am Seitenanfang gesetzte Tabelle etwa mit

```
\vpageref[am Seitenanfang wiedergegebene Tabelle]%  
[Tabelle]{tab:foo}
```

erfolgen. Erscheint die Tabelle erst auf der folgenden Seite, wird mit

```
Tabelle \vpagereftab:foo
```

auf sie verwiesen. Auf der aktuellen Seite wird hingegen

```
am Seitenanfang wiedergegebene Tabelle
```

verwendet.

Gelegentlich möchte man nicht auf eine einzelne Abbildung, Formel oder Seite verweisen, sondern auf mehrere aufeinander folgende Abbildungen, Formeln oder einen Seitenbereich. `varioref` stellt dem Anwender auch für diesen Fall Befehle zur Verfügung:

```
\vrefrange[<Text>]{<Erste Marke>}{<Letzte Marke>}  
\vpagerefrange[<Text>]{<Erste Marke>}{<Letzte Marke>}  
\vpagerefrange*[<Text>]{<Erste Marke>}{<Letzte Marke>}
```

Der erste Befehl `\vrefrange` verweist auf eine Folge von Marken, beginnend mit `<Erste Marke>` und endend mit `<Letzte Marke>`. Er verhält sich ansonsten wie der Befehl `\vref` zum Verweis auf eine einzelne

Marke: Er ignoriert vom Anwender angegebenen Leerraum vor `\vrefrange` und fügt einen eigenen Abstand ein. Dem Verweis auf die Marken wird durch einen Verweis auf die Seite(n) ergänzt, auf denen die Marken zu finden sind. Für den Verweis auf den zugehörigen Seitenbereich wird der Befehl `\vpagerefrange` verwendet, den der Anwender auch direkt zum Verweis auf einen Seitenbereich verwenden kann. Liegen sämtliche Ziele der Verweise auf der aktuellen Seite, wird der Text von `\ref{current}` für den Verweis verwendet. Bei Angabe des optionalen Arguments `<Text>` wird der angegebene Text statt `\ref{current}` gesetzt. Liegen alle Ziele der Verweise auf einer einzigen Seite, verhält sich `\vpagerefrange` exakt wie der Befehl `\pageref`. Der Unterschied zwischen `\vpagerefrange` und `\vpagerefrange*` ist wieder die Behandlung von Leerraum.

Für den Verweis auf eine Folge von Marken und einen Seitenbereich werden die mit den beiden folgenden Befehlen definierten Texte verwendet:

```
\reftextlabelrange{<Erste Marke>}{<Letzte Marke>}
```

Verweis auf aufeinander folgende Marken

Voreinstellung: `\ref{#1} bis~\ref{#2}`

```
\reftextpagerange{<Erste Marke>}{<Letzte Marke>}
```

Verweis auf einen Seitenbereich

Voreinstellung: auf Seiten~`\pageref{#1}--\pageref{#2}`

Ein Verweis auf eine Serie von Abbildungen könnte beispielsweise durch

```
... siehe Abbildungen \vrefrange{fig:a}{fig:c} ...
```

erfolgen. Liegen alle Abbildungen auf der aktuellen Seite, wird

```
... siehe Abbildungen 11 bis 13 auf dieser Seite ...
```

gesetzt. Werden alle Abbildungen auf der folgenden Seite gesetzt, erscheint in der Ausgabe der Verweis

```
... siehe Abbildungen 11 bis 13 auf der nächsten Seite ...
```

Entsprechend wird ein Verweis behandelt, bei dem sämtliche Abbildungen auf der vorhergehenden Seite liegen.

Erscheinen die Abbildungen hingegen auf unterschiedlichen Seiten, wird stets auf einen Seitenbereich verwiesen, selbst wenn die Abbildungen komplett auf der aktuellen und der Folgeseite gesetzt werden:

```
... siehe Abbildungen 11 bis 13 auf Seiten 15–16 ...
```

Korrekturen am zu setzenden Text können in den meisten Fällen durch Redefinition der verwendeten `\ref`-Befehle vorgenommen werden.

Für den Fall, dass eine Anpassung der `\ref`-Befehle nicht ausreicht, stellt `\vref` den Befehl

```
\vrefpagemum{<Befehlsname>}{<Marke>}
```

mit einer durch `\label` definierten `<Marke>` zur Verfügung. `<Befehlsname>` ist ein vom Anwender vorzugebender Befehl, dem die zu `<Marke>` gehörende Seite übergeben wird. Der als Argument angegebene Befehl kann – muss aber nicht – mit einem `\newcommand`-Befehl vereinbart werden. Das folgende Beispiel basiert auf der Dokumentation und definiert einen Befehl zum Verweis auf zwei Abbildungen.

```
\newcommand{\reftwo}[2] {%
  \vrefpagemum{\start}{#1}%
  \vrefpagemum{\ende}{#2}%
  \ifthenelse{\equal{\start}{\ende}}{%
    {\ref{#1}} und \ref{#2} \vpageref{#1}%
    {\ref{#1}} \vpageref{#1} und \ref{#2} \vpageref{#2}%
  }
}
```

Nach dieser Definition kann `\reftwo` etwa zum Verweis auf zwei Abbildungen in der Form

... die Abbildungen `\vrefrange{fig:a}{fig:c}` ...

verendet werden. Werden beide Abbildungen auf einer Seite – beispielsweise der folgenden – gesetzt, wird im Dokument in der Form

... die Abbildungen 11 und 13 auf der nächsten Seite ...

verwiesen. Liegen sie auf verschiedenen Seiten, etwa auf der nächsten und übernächsten Seite, lautet der Verweis:

... die Abbildungen 11 auf der nächsten Seite und 13 auf Seite 16 ...

Fehlermeldungen

Abschließend muss noch auf die Fehlermeldungen des Pakets eingegangen werden. Ursache für Probleme können sein, dass Verweise auf Seiten mit den vorgestellten Befehlen aus mehreren Worten bestehen und damit

Seitenumbrüche im Verweistext möglich sind. Bei Verweisen auf entfernte Seiten mit Angabe der Seitennummer ist ein Seitenumbruch im Verweistext kein Problem. Anders ist die Situation, wenn es sich um einen Verweis auf die aktuelle Seite oder benachbarte Seiten handelt. Beispiele für derartige Fälle sind

Tabelle 5 auf die- *<Seitenumbruch>* ser Seite

oder

Tabelle 5 auf der nächsten *<Seitenumbruch>* Seite.

Da es kaum möglich ist, für alle derartigen Fälle eine brauchbare Lösung zu finden, prüft das Paket, ob innerhalb des Verweistexts ein Seitenumbruch stattfindet, und produziert einen Fehler, wenn dieser Fall eintritt. Durch Angabe der Option `draft` oder den Befehl `\vrefwarning` in der Präambel können die Fehler zu Warnungen gemacht werden, so dass die Formatierung nicht angehalten wird. Die Option `final` der ein `\vrefs-howerrors`-Befehl in der Präambel entspricht, was auch die Voreinstellung ist, stellt hingegen sicher, dass die Formatierung unterbrochen wird, wenn ein Seitenumbruch in einen Verweistext fällt.

Das Problem lässt sich dadurch lösen, dass an diesen Stellen normale `\ref-` oder `\pageref-`Befehle verwendet werden. Auch das Umformulieren des Texts kann das Problem gegebenenfalls vermeiden. Diese Änderungen sollten allerdings erst vorgenommen werden, wenn die eigentliche Arbeit am Text abgeschlossen ist. Bis dahin wird man in der Regel die Fehlermeldungen abschalten.

9.22 Paket verbatim

Dieses Paket von Dr. Rainer Schöpf, Bernd Raichle und Chris Rowley verbessert die Handhabung der `verbatim`-Umgebung und stellt außerdem eine Umgebung für Kommentare zur Verfügung. Das Setzen einer ganzen Datei im `verbatim`-Modus ist ebenfalls möglich.

Die $\text{\LaTeX}$ -Standard-Umgebungen `verbatim` und `verbatim*` – der Unterschied besteht in der Hervorhebung der Leerzeichen durch `\_` – haben zwei Nachteile, die sich aber selten bemerkbar machen:

1. Andere Umgebungen erlauben Leerraum (engl. *white space*) zwischen `\begin` bzw. `\end` und dem in geschweifte Klammern eingeschlossenen Namen der Umgebung. Beim Befehl `\end{verbatim}` und seiner gesternten Form ist hingegen kein Leerraum zugelassen.

2. $\text{\TeX}$ muss den gesamten Inhalt einer `verbatim`-Umgebung im Speicher halten und kann erst mit der Ausgabe beginnen, wenn das Ende der Umgebung erreicht ist. Bei sehr großen `verbatim`-Umgebungen kann es daher passieren, dass die Formatierung mit einem Fehler abbricht, da nicht genügend Speicher zur Verfügung steht.

Die Neuimplementierung der Umgebungen in diesem Paket mindert den ersten Nachteil und vermeidet den zweiten. Der erste Nachteil wird insofern vermindert, als zwischen `\end` und `{verbatim}` bzw. `{verbatim*}` *Leerzeichen* ausdrücklich erlaubt werden. Ein Zeilenumbruch ist aber weiterhin verboten.

Die (naheliegende) Einschränkung, dass die Umgebungen `verbatim` bzw. `verbatim*` nicht geschachtelt werden dürfen, bleibt bestehen. Allerdings kann eine `verbatim*`-Umgebung in einer `verbatim`-Umgebung dargestellt werden und umgekehrt.

Nach dem Laden des Pakets mit

```
\usepackage{verbatim}
```

ist daher eine `verbatim`-Umgebung der Form

```
9  \begin {verbatim}
10 \begin{verbatim*}
11   Ausgabe im
12   verbatim-Modus
13 \end{verbatim*}
14 \end {verbatim}
```

```
\begin{verbatim*}
  Ausgabe im
  verbatim-Modus
\end{verbatim*}
```

[Buch1/Ergaenz/verbatim.tex]

zulässig und wird korrekt formatiert. Beim Schreiben dieses Buchs war übrigens festzustellen, dass die verwendete Kile-Version `\end {verbatim*}` nicht erkannte und die Umgebung als nicht abgeschlossen ansah.

Bei den neu implementierten Umgebungen `verbatim` und `verbatim*` entfällt die Beschränkung der Größe durch den verfügbaren Speicher. Der Preis für das Wegfallen des zweiten Nachteils ist allerdings eine Inkompatibilität mit der ursprünglichen Version. Dort kann nach `\end{verbatim}` bzw. `\end{verbatim*}` in der gleichen Zeile Text folgen, der von $\text{\LaTeX}$ normal berücksichtigt wird. Die neu implementierten Versionen ignorieren hingegen alles nach `\end{verbatim}` und `\end{verbatim*}` bis zum Zeilenende! Werden Zeichen unter diesen Umständen ignoriert, gibt das Paket eine Warnung aus der Form

LaTeX Warning: Characters dropped after ‘\end{verbatim}’ on input line 68.

Damit ist die Information des Anwenders sichergestellt, und die Zeichen fallen nicht stillschweigend unter den Tisch.

Zusätzlich definiert das Paket zwei Befehle, mit denen ganze Dateien im `verbatim`-Stil gesetzt werden können:

```
\verbatiminput{<Datei>}
\verbatiminput*{<Datei>}
```

Bei `\verbatiminput*` werden Leerzeichen dabei besonders hervorgehoben durch `_`. `<Datei>` ist die externe Datei, deren Inhalt so gesetzt wird, als stünde er komplett in einer `verbatim`- oder `verbatim*`-Umgebung.

Schließlich stellt das Paket noch eine Kommentar-Umgebung zur Verfügung. Anlass dafür ist der Umstand, dass eine `verbatim`-Umgebung den Inhalt liest, ohne ihn zu interpretieren, um ihn also unverändert auszugeben. Wird die Ausgabe aus bestimmten Gründen unterdrückt, hat man mit der Kommentar-Umgebung eine Alternative, die die folgende Form hat:

```
\begin{comment}
  <Kommentar>
\end{comment}
```

`<Kommentar>` kann alles enthalten, was eine `verbatim`-Umgebung enthalten darf. Die einzige Ausnahme ist der Befehl `\end{comment}` (gegebenenfalls auch mit Leerzeichen zwischen `\end` und `{comment}`). Ebenso wie die `verbatim`-Umgebung darf auch die Umgebung `comment` nicht geschachtelt werden.

Die Originaldokumentation zu diesem Paket enthält darüber hinaus eine Reihe von Hinweisen und Beispielen, die allerdings vor allem beim Schreiben eigener Pakete eine Rolle spielen. Einiges kann man auch in normalen Dokumenten verwenden, und diese Teile werden im Folgenden kurz vorgestellt.

Ein wichtiger Vorteil der neu implementierten Umgebung ist die Möglichkeit, eigene (modifizierte) `verbatim`-Umgebungen zu vereinbaren. Dabei lassen sich beispielsweise auch Schriftart oder -größe ändern, die beim Setzen des Inhalts verwendet werden. Befehle wie `\begin{verbatim}` und `\end{verbatim}` dürfen jedoch weder bei der Definition eigener Befehle noch bei der Definition eigener Umgebung verwendet werden und

führen beim Formatieren zu einem Fehler. Nun bildet aber $\LaTeX$ `\begin{<Umgebungsname>}` und `\end{<Umgebungsname>}` auf die Befehle `\<Umgebungsname>` und `\end{<Umgebungsname>}` ab, und diese Befehle können bei eigenen `verbatim`-Umgebungen wiederum verwendet werden. Dann darf die Definition jedoch überhaupt keine `\begin`- und `\end`-Befehle enthalten. Auch andere Umgebungen müssen unter diesen Umständen mit den ihnen zugeordneten Befehlen begonnen und beendet werden.

Besonders interessant ist also, dass bei der Definition einer eigenen `verbatim`-Umgebung dem `\verbatim` weitere Befehle folgen können, die ausgeführt werden, bevor der Inhalt der Umgebung gesetzt wird. Das folgende Beispiel zeigt die Definition einer eigenen `verbatim`-Umgebung mit Umschalten der Schriftart.

```

1  \documentclass{article}
2  \usepackage[T1]{fontenc}
3  \usepackage[ngerman]{babel}
4  \usepackage[dvipdfm]{color}
5  \definecolor{grau}{gray}{.8}
6  \usepackage{verbatim}
7  \setlength{\textwidth}{5cm}
8  \setlength{\parindent}{0cm}
9  \pagestyle{empty}
10 \newenvironment{vrbspec}%
11   {\par\verbatim
12    \itshape\color{grau}}%
13   {\endverbatim}
14 \begin{document}
15 \slshape
16 \begin{verbatim}
17 Text im verbatim-Modus
18 nach \slshape{}.
19 \end{verbatim}
20
21 Die Schrift im Flie"stext
22 ist geneigt!
23
24 \begin{vrbspec}
25 Text in der neu definierten
26 Umgebung vrbspec
27 mit grauer Schrift und
28 kursivem Schriftschnitt
29 (\itshape).
30 Sieht ganz anders aus
31 als 'normales' verbatim.
32 \end{vrbspec}
33 \end{document}

```

Text im verbatim-Modus
nach `\slshape{}`.

Die Schrift im Fließtext ist geneigt!

*Text in der neu definierten
Umgebung vrbspec
mit grauer Schrift und
kursivem Schriftschnitt
(\itshape).
Sieht ganz anders aus
als 'normales' verbatim.*

[Buch1/Ergaenz/verbatim2.tex]

Teil des Pakets ist auch das Beispieldokument `verbstest.tex`. Es zeigt die Verwendung der Befehle und Umgebungen dieses Pakets und kann dem Anwender auch als Testbeispiel dienen.

9.23 Paket xr

Die Befehle `\label`, `\ref` und `\pageref` erlauben Querverweise innerhalb eines Dokuments. Ein Verweis auf mit dem `\label`-Befehl in anderen Dokumenten vereinbarte Ziele von Querverweisen – beispielsweise bei mehrbändigen Büchern – wird von Standard- $\text{\LaTeX}$ nicht unterstützt. Das Paket `xr` von David Carlisle macht es nun möglich, mit `\ref` und `\pageref` auf Stellen in anderen Dokumenten zu verweisen.

Die Befehle `\ref` und `\pageref` verhalten sich wie gewohnt, nur kann die verwendete Marke auch in anderen Dokumenten vereinbart sein. Das Paket `showkeys` zeigt dann auch die externen Marken korrekt an.

Nach dem Laden des Pakets mit

```
\usepackage{xr}
```

müssen *in der Präambel* die zu berücksichtigenden externen Dokumente aufgeführt werden. Für jedes externe Dokument muss dazu ein Befehl

```
\externaldocument[<Präfix>]{<Dokument>}
```

verwendet werden. `<Dokument>` ist dabei der Dateiname eines (Teil-)Dokuments ohne die Erweiterung des Dateinamens. Dabei ist zu beachten, dass die Definitionen der Marken in der zugehörigen `.aux`-Dateien abgelegt werden. Mit `\include` eingebundene Teildokumente haben eine eigene `.aux`-Datei, und da `xr` die `.aux`-Datei nicht vollständig analysiert, muss in diesen Fällen die mit `\include` eingebundene Datei und nicht die Hauptdatei des externen Dokuments angegeben werden.

Die in `\ref`- oder `\pageref`-Befehlen verwendeten Marken müssen weiterhin eindeutig sein, und zwar unter Einbeziehung aller externen Dokumente. Damit man die externen Dokumenten nicht alle prüfen muss, gibt es die Möglichkeit durch Angabe von `<Präfix>`, die Marken zu modifizieren. Im externen Dokument wird jedem Verweis `<Präfix>` vorangestellt und damit eindeutig gemacht.

Mit

```
\externaldocument[X1-]{xyz}
```

wird also allen Verweisen, die sich auf das Dokument `xyz` beziehen, der Präfix `X1-` vorangestellt. Eine mit

```
\label{Kap1}
```

in `xyz` mit vereinbarte Marke, muss dann durch

```
\ref{X1-Kap1}
```

referenziert werden.

Für *<Präfix>* gelten im Übrigen die gleichen Regeln wie bei Referenzen in nur einem Dokument. *Aktive Zeichen*, beispielsweise `:` bei einem französischen Text oder `"` bei einem deutschen Dokument sind in Marken zu vermeiden.

9.24 Paket `xspace`

$\LaTeX$ betrachtet Leerzeichen oder den Übergang auf eine neue Zeile nach einem Befehl ohne Argumente nicht als Wortzwischenraum, sondern als Abgrenzung des Befehls vom nachfolgenden Text. Soll dem Befehl ein Wortzwischenraum folgen, hängt man häufig ein Paar geschweifeter Klammern an. Dieses Paket von David Carlisle bietet dazu eine Alternative. Es wird geladen mit

```
\usepackage{xspace}
```

Fügt man bei der Definition eines Makros am Ende den Befehl

```
\xspace
```

ein, so wird nach dem Befehl automatisch ein Wortzwischenraum eingefügt, falls ihm nicht direkt Satzzeichen

`. ! , : ; ? / ' ) -`

oder Sonderzeichen, genauer ein geschweiftes Klammernpaar oder explizite Leerzeichen

```
{ } ~ \ / \_
```

folgen.

Zu beachten ist, dass das Paket `xspace` erst nach allen Sprachpaketen und sonstigen Paketen geladen werden darf, die Satzzeichen ‚aktiv‘ machen.

Das folgende Beispiel zeigt, wie `xspace` die Eingabe vereinfachen kann:

```

7  \usepackage{xspace}
8  \begin{document}
9  \newcommand{\TK}{\TeX\nische
10     Kom"odie\xspace}
11  \newcommand{\TEX}{\TeX\xspace}
12  Die \TK ist die
13  Mitgliederzeitschrift von
14  DANTE, der deutschsprachigen
15  Anwendervereinigung \TEX e.V.
16  Mitglieder erhalten
17  mehrmals j"ahrlich die \TK,
18  mit \TEX{}nischen Infos.
19  \end{document}

```

Die $\text{\TeX}$ nische Komödie ist die Mitgliederzeitschrift von DANTE, der deutschsprachigen Anwendervereinigung $\text{\TeX}$ e.V. Mitglieder erhalten mehrmals jährlich die $\text{\TeX}$ nische Komödie, mit $\text{\TeX}$ nischen Infos.

[Buch1/Ergaenz/xspace.tex]

Grafik, Bilder und Farben

Seit dem Einzug von Farbdruckern und farbfähigen Druckmaschinen ist es heute kein Problem mehr, farbige Dokumente zu drucken. Im Laufe der Zeit wurden daher auch für $\text{\LaTeX}$ verschiedene Pakete geschrieben, die die Verwendung von Farben und das Einbinden von Bildern unterstützen. Die hier behandelten Pakete von David Carlisle zählen zu den Grundpaketen, bei denen man davon ausgehen kann, dass sie Teil jeder $\text{\LaTeX}$ -Installation sind.

Pakete: color, graphics, graphicx, epsfig, trig, keyval, pstcol, lscape

color, graphics

Von zentraler Bedeutung sind die Pakete color für farbige Schriften und Hintergründe und graphics zum Einbinden von Bildern und zur grafischen Manipulation von Textpassagen.

graphicx

Beim Paket graphicx handelt es sich um ein Interface zum Paket graphics, das eine stärker an $\text{\LaTeX}$ orientierte Form der Befehle zur Verfügung stellt.

epsfig

Das Paket epsfig erlaubt das Einbinden von PostScript-Bildern mit den Befehlen des (inzwischen veralteten) Pakets psfig.

trig, keyval

Die Pakete trig und keyval sind eher für Entwickler eigener Pakete von Interesse. Sie werden daher hier nur kurz behandelt.

pstcol

Ein sehr mächtiges Paket zum Erstellen von PostScript-Grafiken ist *pstricks* von Timothy Van Zandt. Die dort verwendeten Befehle für Farbdefinitionen sind jedoch leider nicht kompatibel zu den entsprechenden Farbdefinitionen im Paket *color*. Das Paket *pstcol* erlaubt bei Verwendung der beiden Pakete *color* und *pstricks* die Austauschbarkeit der Farbdefinitionen. Damit können Farben unabhängig davon verwendet werden, ob sie mit den Befehlen des Pakets *color* oder denen des Pakets *pstricks* vereinbart wurden.

lscope

Das Paket *lscope* stellt eine Umgebung `landscape` zur Verfügung, in der der Textbereich von Seiten um 90 Grad gedreht ist. Kopf- und Fußzeilen werden nicht gedreht und erscheinen daher an der gewohnten Stelle.

Zu den Paketen *epsfig*, *trig*, *keyval*, *pstcol* und *lscope* dürfte die Kurzbeschreibung ausreichend sein. Die Pakete *color* und *graphics/graphicx* stellen jedoch eine Reihe von Befehlen zur Verfügung, die im Folgenden genauer betrachtet werden.

Zuvor muss aber ein anderer Aspekt betrachtet werden: Die Notwendigkeit einer Anpassung an die verschiedenen Ausgabegeräte.

10.1 Gerätetreiber

Bedauerlicherweise gibt es keine Standards für die Weitergabe von Anweisungen an die verschiedenen Ausgabegeräte, nicht einmal für einzelne Geräteklassen wie Drucker oder Plotter. Aus diesem Grund müssen die Befehle, die die grafischen Fähigkeiten der Ausgabegeräte ansprechen, in gerätespezifische Steueranweisungen umgesetzt werden. Diese Umsetzung führen die Gerätetreiber durch. Die Auswahl der Gerätetreiber erfolgt durch Angabe ihres Namens als Option beim Laden der Pakete oder durch eine Anweisung in der Konfigurationsdatei des jeweiligen Pakets.

Im Zusammenhang mit den Standard-Grafikpaketen stehen die in Tabelle 10.1 aufgeführten Treiber für verschiedene Ausgabegeräte zur Verfügung.

Zusätzlich können die in Tabelle 10.2 genannten – mangels entsprechender Geräte nicht vollständig ausgetesteten – Treiber für ältere oder ungewöhnliche Ausgabegeräte generiert und mit dem Befehl

Tabelle 10.1: Treiber für die Standard-Grafikpakete

Option	für
<i>Previewer:</i>	
xdvi	xdvi unter Unix (identisch mit dvips)
dviwin	dviwin von Hippocrates Sendoukas
dviwindo	dviwindo von Inc. Y&Y
<i>PostScript:</i>	
dvips	dvips von Tom Rokicki
dvipsone	dvipsone von Inc. Y&Y
<i>PDF:</i>	
dvipdf	dvipdf
dvipdfm	dvipdfm
pdfTEX	pdfTEX von Hàn Thê Thành
<i>Spezielle T_EX-Distributionen:</i>	
ozTEX	OzTEX ab Version 1.42 von Andrew Trevorrow
emTEX	emTEX von Eberhard Mattes
textures	Textures von Inc. Blue Sky Research
pctexps	PTI Laser/PS von Inc. Personal T _E X
pctex32	von Inc. Personal T _E X für 32 bit Windows
pctexwin	von Inc. Personal T _E X für Windows
pctexhp	PTI Laser/HP von Inc. Personal T _E X
truetex	truetex von Richard Kinch
tcidvi	von Inc. Scientific Word/Workplace
vtex	VT _E X ab Version 6.3 von Inc. MicroPress

```
\ExecuteOption{<Treiber>}
```

aktiviert werden. Diese Treiber werden von einer normalen T_EX-Installation nicht automatisch zur Verfügung gestellt.

Tabelle 10.2: Treiber der Standard-Grafikpakete für ältere oder ungewöhnliche Ausgabegeräte

Option	für
dvalw	dvalw von Nelson Beebe
dvilaser	dvilaser/ps von Inc. Arbortext
psprint	psprint von Andrew Trevorrow
dvitops	dvitops von James Clark
dvi2ps	dvi2ps
pubps	PUBps von Inc. Arbortext
ln	ln03 von B. Hamilton Kelly

Die Treiber unterscheiden sich – bedingt durch die unterschiedlichen Ausgabegeräte – in ihren Fähigkeiten, beispielsweise beim Drehen und Skalieren von Bildern und Textpassagen sowie in den zur Verfügung stehenden Farben. Es gibt auch Unterschiede bei den Bildformaten, die eingebunden werden können. Eine Übersicht über die Fähigkeiten der Treiber geben die Tabellen 10.3 und 10.4.

Tabelle 10.3: Fähigkeiten spezieller Gerätetreiber

Treiber	Drehen	Skalieren	Spiegeln	Farbe	Bildformate
dvialw					.ps
dvilaser					.ps
psprint					.ps
dvitops	✓				.ps
dvi2ps					.ps
pubps	✓				kein Einbinden von Bildern
ln					nur Spezialformate

10.2 Paket color

Mit Hilfe dieses Pakets ist es möglich, den Hintergrund einer oder mehrerer Seiten einzufärben oder Textpassagen farbig zu setzen und zu hinterlegen.

Die Festlegung von Farbwerten ist in verschiedenen Formen möglich. Neben den Farbmodellen RGB, CMYK, HSB und Grauwerten können auch zuvor definierte Farbnamen verwendet werden. Das Farbmodell unter Verwendung vordefinierter Farbnamen soll auch als Beispiel für lokale zusätzliche Farbmodelle dienen. Da das Hinzufügen weiterer Farbmodelle Eingriffe in das Paket color erfordert, sind dazu tiefere Kenntnisse von $\text{\LaTeX}$ eine wesentliche Voraussetzung.

Die Farbmodelle werden mit den folgenden Bezeichnungen angesprochen und haben die angegebene Bedeutung:

- rgb** Grundfarben rot, grün und blau
Die Anteile der Grundfarben liegen zwischen 0 (kein Anteil an der gemischten Farbe) und 1 (maximaler Anteil) und werden als durch Kommas getrennte Liste der Anteile der drei Grundfarben in der Reihenfolge wie oben angegeben.
Bsp.: 0, 0.3, 0.62

Tabelle 10.4: Fähigkeiten der allgemeinen Gerätetreiber

Treiber	Drehen	Skalieren	Spiegeln	Farbe	Bildformate
dvips	✓	✓	✓	✓	.ps, .bmp ^a , .msp ^b , .pcx ^c , .pict ^d , .pntg ^e
dvipdf	✓	✓	✓	✓	.ps, .bmp ^a , .jpg ^f , .msp ^b
xdvi	✓	✓	✓	✓	.ps, .bmp ^a , .msp ^b , .pcx ^c , .pict ^d , .pntg ^e
dviwin					.ps, .bmp ^a
dviwindo	✓	✓	✓	✓	.ps, .clp ^g , .msp ^b , .pcx ^c , .tif ^h , .wmf ⁱ
pdftex	✓	✓	✓	✓	.pdf, .jpg ^f , .mps ^j , .png ^k , .tif ^h
dvipsone	✓	✓	✓	✓	.ps, .clp ^g , .msp ^b , .pcx ^c , .tif ^h , .wmf ⁱ
dvipdfm	✓	✓	✓	✓	.ps, .jpeg/.jpg ^f , .png ^k
oxtex	✓		✓	✓	.ps, .bmp ^a , .msp ^b , .pcx ^c , .pict ^d , .pntg ^e
emtex					.bmp ^a , .msp ^b , .pcx ^c
textures	✓	✓	✓	✓	.ps, .rps ^l , .pict ^d , .pic- ture ^m
pctexps	✓	✓	✓	✓	.ps
pctex32	✓	✓	✓	✓	.ps, .bmp ^a , .wmf ⁱ
pctexwin					.ps, .bmp ^a , .wmf ⁱ
pctexhp					.pcl ⁿ
truetex				✓	.ps, .bmp ^a , .tif ^h
tcidvi				✓	.ps, .bmp ^a
vtex	✓	✓	✓	✓	.ps, .bmp ^a , .emf ^o , .gif ^p , .jpeg/.jpg ^f , .mps ^j , .pcx ^c , .png ^k , .tga/.targa ^q , .tiff/.tif ^h , .wmf ⁱ

^a *.bmp für Microsoft Windows Bitmap^b *.msp für Microsoft Windows Paint (Vorläufer von .bmp)^c *.pcx für ZSoft Paintbrush^d *.pict für MacIntosh PICT Format^e *.pntg für Apple MacIntosh MacPaint File (Bitmap graphics B&W)^f *.jpg für JPEG File Interchange Format^g *.clp für PC Paint Bitmap graphics (Clip art)^h *.tif für Tagged Image File Formatⁱ *.wmf für Windows Metafile^j *.mps für METAPost-Ausgaben^k *.png für Portable Network Graphics^l *.rps für PostScript (nicht encapsulated)^m *.picture für type PICTUREⁿ *.pcl für HP PCL^o *.emf für Windows Enhanced Metafile^p *.gif für Graphics Interchange Format^q *.targa für Targa Image File

cmyk Grundfarben cyan, magenta, gelb und schwarz
Die Farbanteile liegen ebenfalls zwischen 0 (kein Anteil an der gemischten Farbe) und 1 (maximaler Anteil) und werden als durch Kommas getrennte Liste der Anteile der vier Grundfarben in der Reihenfolge wie oben angegeben.

Bsp.: 0, 0.35, 0.6, 1

hsb (Hue – Saturation – Brightness)

Auch bei diesem Farbmodell werden Farben durch eine Liste von drei Werte zwischen 0 und 1 festgelegt.

Mit dem ersten Wert (Hue) werden dabei die Farbanteile festgelegt, wobei 0 einem reinen Rot, $\frac{1}{3}$ einem reinen Grün, $\frac{2}{3}$ einem reinen Blau und 1 wieder dem reinen Rot entspricht. Die Farben sind also in einem Farbkreis angeordnet.

Der zweite Wert (Saturation) legt die Farbsättigung fest. Ein Wert von 0 bedeutet, dass die Farbanteile vollständig unterdrückt werden und damit ein Grauwert festgelegt wird. Bei einem Wert von 1 werden hingegen die Farbanteile voll wiedergegeben. Zwischenwerte entsprechen einer Mischung des reinen Farbwerts mit weißem Licht.

Schließlich legt der dritte Wert die Helligkeit der Farbe fest. Ein Wert von 0 wählt – unabhängig von den anderen Werten – die Farbe Schwarz aus. Der Wert 1 ergibt die maximale Helligkeit, die unter Beachtung der beiden vorhergehenden Werte möglich ist.

Bsp.: 0.3333, 1, 0.5

gray Man legt Grauwerte durch einen einzelnen Wert zwischen 0 (Schwarz) und 1 (Weiß) fest

Bsp.: 0.75

named Vordefiniert sind die 68 in den Paketen `colordvi` und `foiltex` von Jim Hafner definierten Farbnamen.

Ein wesentlicher Vorteil dieses Farbmodells besteht darin, dass die geräteabhängige Farbzusammensetzung in einer Datei `color.pro` festgelegt werden kann und damit leicht auszutauschen ist.

Nicht alle Farbmodelle können mit jedem Ausgabegerät (Treiber) verwendet werden, beispielsweise sind Farbangaben im hsb-Modell bei PostScript aber nicht bei PDF möglich.

Häufig wird das Paket einschließlich einer Treiberangabe geladen. In diesem Fall hat der Befehl die folgende Form:

```
\usepackage[<Treiber>[,<Weitere Option(en)>]]{color}
```

Dabei steht *<Treiber>* für einen der in 10.1 behandelten Treiber für das spätere Ausgabegerät. Fehlt die Angabe des Treibers, wird der in der Konfigurationsdatei `graphics.cfg` angegebene Treiber eingesetzt.

Daneben stehen die folgenden paketspezifischen Optionen für besondere Fälle zur Verfügung:

monochrome	Bei der Weitergabe von Farbbefehlen an den Gerätetreiber werden gelegentlich durch diese Befehle Abstände an diesen Stellen verändert. Mit dieser Option lässt sich in diesen Fällen eine einfarbiger (Test-)Ausdruck mit exakt den gleichen Abständen wie bei einer Farbausgabe erstellen.
nodvipsnames	Die Angabe dieser Option unterdrückt die Definition der 68 im Treiber <code>dvips</code> vordefinierten Farbnamen des benannten Farbmodells. Dadurch lässt sich etwas Platz sparen, der zur Definition anderer Befehl zur Verfügung steht.
dvipsnames	Diese Option stellt die 68 im Treiber <code>dvips</code> vordefinierten Standardfarben des benannten Farbmodells den Treibern zur Verfügung, in denen sie nicht definiert werden.
usenames	Wenn diese Option angegeben ist, können sämtliche definierten Farbnamen im benannten Farbmodell verwendet werden.

In der Regel erfolgt die Festlegung einer zu verwendenden Farbe über einen Farbnamen. Die Farbnamen werden in der Datei `dvipsnam.def` über das Farbmodell **cm^yk** definiert. Abbildung 10.1 zeigt die vordefinierten Farbnamen zusammen mit der Farbe. Beim Erstellen der Tabelle wurde das Paket `color` mit den Optionen `dvips` und `usenames` geladen.

Neben den vordefinierten Farben können auch eigene Farben vereinbart werden:

```
\definecolor{<Name>}{<Farbmodell>}{<Farbspez>}
```

Dabei steht *<Name>* für den (neu) zu vergebenden Farbnamen und *<Farbmodell>* für eines der oben aufgeführten (oder andernfalls lokal definierten) Farbmodelle. *<Farbspez>* ist die dem Farbmodell entsprechende Festlegung einer Farbe.

Farbname	Farbe	Farbname	Farbe
GreenYellow		RoyalPurple	
Yellow		BlueViolet	
Goldenrod		Periwinkle	
Dandelion		CadetBlue	
Apricot		CornflowerBlue	
Peach		MidnightBlue	
Melon		NavyBlue	
YellowOrange		RoyalBlue	
Orange		Blue	
BurntOrange		Cerulean	
Bittersweet		Cyan	
RedOrange		ProcessBlue	
Mahogany		SkyBlue	
Maroon		Turquoise	
BrickRed		TealBlue	
Red		Aquamarine	
OrangeRed		BlueGreen	
RubineRed		Emerald	
WildStrawberry		JungleGreen	
Salmon		SeaGreen	
CarnationPink		Green	
Magenta		ForestGreen	
VioletRed		PineGreen	
Rhodamine		LimeGreen	
Mulberry		YellowGreen	
RedViolet		SpringGreen	
Fuchsia		OliveGreen	
Lavender		RawSienna	
Thistle		Sepia	
Orchid		Brown	
DarkOrchid		Tan	
Purple		Gray	
Plum		Black	
Violet		White	

Abbildung 10.1: Farbnamen im Paket color

Vordefinierte – und selbst definierte – Farbnamen können verwendet werden, um die Farbe des Texts oder die Hintergrundfarbe festzulegen. Wie auch bei den Schriftumschaltungen kann man auch hier zwischen zwei verschiedenen Formen wählen. Der Befehl

```
\color{<Farbe>}
\color[<Farbmodell>]{<Farbspez>}
```

schaltet ab der Stelle, an der er steht, für den folgenden Texts in die angegebene *<Farbe>*. In der zweiten Form schaltet er in die durch das *<Farbmodell>* und die dem Farbmodell entsprechende *<Farbspez>* festgelegte Farbe (Angaben wie beim Befehl `\definecolor`) bis zum Ende der aktuellen Gruppe um. Er entspricht damit einem Befehl zur Schriftumschaltung in der Form `\slshape`.

Einer Schriftumschaltung mit einem Befehl der Form `\textsl{<Text>}` entspricht

```
\textcolor{<Farbe>}{<Text>}
\textcolor[<Farbmodell>]{<Farbspez>}{<Text>}
```

Er setzt den angegebenen *<Text>* in der vorgegebenen *<Farbe>* und schaltet danach zurück in die bisherige Textfarbe. In der zweiten Form können wiederum die Argumente des `\definecolor`-Befehls zur Festlegung der Farbe verwendet werden.

Eine Farbangabe durch *<Farbmodell>* und *<Farbspez>* ist nur in besonderen Fällen sinnvoll. Insbesondere ist es mit dieser Form der Farbbefehle möglich, die im Farbmodell `named` vordefinierten Farben unter Angabe des Farbmodells mit ihrem Namen anzusprechen. Um die ständige Wiederholung des Farbmodells zu vermeiden, sollte man aber trotzdem eigene Farbnamen mit Hilfe des Befehls `\definecolor` vergeben.

Das folgende Beispiel zeigt neben den Befehlen `\color` und `\textcolor` auch den `\definecolor`-Befehl.

```
9  \definecolor{G1}{gray}{.25}
10 \definecolor{G3}{gray}{.75}
11 \color{G3}
12  Farbe: gray 0.75\\
13 \color[gray]{.5}
14  Farbe: gray 0.50\\
15 \textcolor{G1}%
16  {Farbe: gray 0.25}\\
17 \textcolor[rgb]{0,0,0}%
18  {Farbe: rgb 0,0,0}
```

Farbe: gray 0.75
 Farbe: gray 0.50
 Farbe: gray 0.25
 Farbe: rgb 0,0,0

[Buch1/Grafik/graycolor.tex]

Ein weiterer Befehl in diesem Paket erlaubt es, den Seitenhintergrund einzufärben. Dabei ist aber unbedingt zu beachten, dass das Einfärben des Seitenhintergrunds den Farbverbrauch *maximiert*. Der Befehl hat die beiden Formen

```
\pagecolor{<Farbe>}
\pagecolor[<Farbmodell>]{<Farbspez>}
```

Die Argumente haben die gleiche Bedeutung wie beim `\color`-Befehl. Da die Hintergrundfarbe global festgelegt wird, ist ein Befehl `\pagecolor{white}` erforderlich, um wieder den ‚normalen‘ Seitenhintergrund herzustellen.

```
7 \usepackage[dvips]{color}
8 \begin{document}
9 \pagecolor[hsb]{0, 0, 0.92}
10 Die Hintergrundfarbe ist ein Grauwert, der
11 mit dem Farbmodell HSB festgelegt wurde.
12 Arbeitet nur mit Grafiktreiber dvips.
13 \end{document}
```

Die Hintergrundfarbe ist ein Grauwert, der mit dem Farbmodell HSB festgelegt wurde; arbeitet nur mit Grafiktreiber dvips.

Bei diesem eingebundenen Beispiel werden Seitenbreite und -höhe ignoriert. Der Befehl färbt wirklich die Seite vollständig ein, unabhängig vom gewählten Papierformat.

Eine letzte Gruppe von Befehlen erlaubt es, Textpassagen farbig zu hinterlegen und ferner farbig einzurahmen. Die dafür verwendeten Befehle entsprechen den Befehlen `\mbox{<Text>}` bzw. `\fbox{<Text>}`:

```
\colorbox{<Hintergrund>}{<Text>}
\colorbox[<Farbmodell>]{<H-Spez>}{<Text>}
\fcolorbox{<Rahmen>}{<Hintergrund>}{<Text>}
\fcolorbox[<Farbmodell>]{<R-Spez>}{<H-Spez>}{<Text>}
```

Der `\colorbox`-Befehl hinterlegt den `<Text>` mit der als `<Hintergrund>` angegebenen Farbe (erste Form). In der zweiten Form des Befehls kann wieder das `<Farbmodell>` angegeben und die Farbe durch die ihm entsprechende Farbspezifikation `<H-Spez>` festgelegt werden.

Beim `\fcolorbox`-Befehl kommt zur Hintergrundfarbe noch die Rahmenfarbe `<Rahmen>` hinzu, oder in der zweiten Form die dem `<Farbmodell>` entsprechende Farbspezifikation `<R-Spez>` für den Rahmen. Die Rahmenfarbe wird in beiden Formen *vor* der Farbe des Hintergrunds angegeben.

Der Abstand des Rahmens vom umschlossenen `<Text>` bzw. des Randes des Hintergrunds vom hinterlegten `<Text>` wird wie beim `\fbox`-Befehl durch die Längenangabe `\fboxsep` festgelegt. In gleicher Weise wird durch die Längenangabe `\fboxrule` die Breite (Strichstärke) des Rahmens vorgegeben.

```

9  \definecolor{G1}{hsb}{0, 0, 0.92}
10 \definecolor{G2}{gray}{0.65}
11 Text in einer \colorbox{G1}
12   {\texttt{\textbackslash colorbox}}
13 wird in der aktuellen Textfarbe gesetzt.
14 Ohne
15   {\color{white}
16   \colorbox[rgb]{.75,.75,.75}{Farbdefinition}}
17 geht's auch.
18
19 Entsprechend auch Boxen mit Rahmen:
20 \fcolorbox{G2}{G1}{Rahmen und Hintergrund}.
21 Ohne Farbdefinition wie in
22 \textcolor{white}{\setlength{\fboxrule}{2.4pt}%
23 \fcolorbox[gray]{.85}{.5}{Wei"ser Text}}
24 muss man mehr schreiben.
```

Text in einer `\colorbox` wird in der aktuellen Textfarbe gesetzt. Ohne Farbdefinition geht's auch.

Entsprechend auch Boxen mit Rahmen: `Rahmen und Hintergrund`. Ohne Farbdefinition wie in `Weißer Text` muss man mehr schreiben.

Das Beispiel zeigt auch, wie die Breite des Rahmens durch Ändern der Länge `\fboxrule` verändert werden kann.

Da $\text{\TeX}$ keine Farben kennt, müssen die Farbbefehle von den Gerätetreibern umgesetzt werden. Einige Gerätetreiber verfügen beispielsweise nicht über einen Farb-Stack und können verschachtelte Farbbefehle oder

Farbbefehle in gleitenden Umgebungen wie `table` oder `figure` nicht korrekt ausführen. Andere Treiber sind nicht in der Lage, sich Farbwechsel über einen Seitenumbruch hinweg zu merken, und setzen dann einen farbigen Text auf der neuen Seite in schwarz fort. Die Gerätetreiber für PostScript und PDF verhalten sich aber korrekt.

Ein weiteres Problem ist hingegen allen Treibern gemeinsam: Durch die Notwendigkeit, Farbbefehle an den Gerätetreiber weiterzugeben, kann es vorkommen, dass Abstände an Stellen verändert werden, an denen die Farbe wechselt. Die Angabe der Option `monochrome` ermöglicht es, in diesen Fällen einen einfarbigen (Test-)Ausdruck mit exakt den gleichen Abständen wie bei der Farbausgabe zu erstellen.

10.3 Pakete `graphics` und `graphicx`

Die Fähigkeiten der beiden Pakete sind identisch; sie unterscheiden sich lediglich in der Form der Befehle. Aus diesem Grund werden beide Pakete gemeinsam behandelt. Die Pakete stellen Befehle zur Verfügung, mit denen Bilder eingebunden und Ausgabeelemente grafisch manipuliert werden können.

Bei Verwendung des Pakets `graphicx` kann die Wirkung der Befehle durch Angabe von Optionen beeinflusst werden. Beim Paket `graphics` verwendet man zu diesem Zweck globale Optionen und kombiniert mehrere Befehle.

Wie beim Paket `color` ist auch bei diesen beiden Paketen die wichtigste Option die Angabe des zu verwendenden Treibers. Dann haben die Befehle zum Laden der Pakete die folgende Form:

```
\usepackage[<Treiber>[, <Weitere Option(en)>]]{graphics}
\usepackage[<Treiber>[, <Weitere Option(en)>]]{graphicx}
```

`<Treiber>` ist wieder einer der in 10.1 behandelten Treiber für das spätere Ausgabegerät. Fehlt die Angabe des Treibers, wird der in der Konfigurationsdatei `graphics.cfg` angegebene Treiber eingesetzt.

Zusätzlich kennen die beiden Pakete die folgenden paketspezifischen Optionen:

draft unterdrückt alle Ausgaben, die auf den Grafikfähigkeiten der beiden Pakete beruhen, insbesondere die Ausgabe von Bildern. Der zur Ausgabe benötigte Platz wird jedoch freigehalten, und bei eingebundenen Bildern wird statt des Bilds der Dateiname in einem der Bildgröße entsprechenden Rahmen ausgegeben.

final ist das Gegenteil der Option `draft`. Da sie ohnehin eingeschaltet ist, wenn die Option nicht angegeben ist, wird sie in der Regel nur benötigt, um eine globale Option `draft` in der `\documentclass` zu überschreiben.

hiderotate

unterdrückt die Ausgabe von gedrehtem Text. Damit lassen sich beispielsweise Probleme mit Previewern vermeiden, die gedrehte Texte nicht darstellen können.

hidescale

unterdrückt die Ausgabe skalierten Texte, etwa wenn der Previewer Texte nicht skalieren kann.

hiresbb

Eingebundene EPS-Dateien enthalten in der Regel eine Größenangabe der Form:

```
%%BoundingBox: <Lx> <Ly> <Rx> <Ry>
```

Die Koordinaten `<Lx>` und `<Ly>` der linken unteren und `<Rx>` und `<Ry>` der rechten oberen Ecke sind ganzzahlig und lassen daher nur eine Auflösung von 1 bp (1/72 in) zu. Ist eine genauere Größenangabe in der Form

```
%%HiResBoundingBox: <Lx> <Ly> <Rx> <Ry>
```

möglich, bei der die Koordinaten der Eckpunkte auch Nachkommastellen haben können. Die Option `hiresbb` bewirkt, dass nach einer Größenangabe in der hochauflösenden Form gesucht werden soll.

Die beiden Pakete ermöglichen das Drehen von Ausgabeelementen um beliebige Winkel, das freie Skalieren von Ausgabeelementen und das Einbinden von Bildern. Das Drehen um beliebige Winkel und das Skalieren ist allerdings je nach dem gewählten Ausgabegerät eingeschränkt oder sogar ausgeschlossen (vgl. Tabelle 10.4).

10.3.1 Drehen von Ausgabeelementen

Alle Ausgabeelemente, die in eine Box gepackt werden können, lassen sich um einen angegebenen Winkel gegen den Uhrzeigersinn drehen. Das als Argument angegebene Ausgabeelement wird in eine Box gesetzt. Die Drehung erfolgt um den Referenzpunkt dieser Box, wenn er nicht ausdrücklich abweichend festgelegt wird.

Der Befehl zum Drehen hat die Form:

```
graphics: \rotatebox{<Winkel>}{<Text>}
graphicx: \rotatebox[<Option(en)>]{<Winkel>}{<Text>}
```

Der Drehwinkel *<Winkel>* wird in der Regel in Grad gegen den Uhrzeigersinn angegeben. Bei Bedarf lässt sich die Drehrichtung umstellen. Auch eine Umstellung der Winkelangabe auf das Bogenmaß ist möglich. *<Text>* ist das zu drehende Ausgabeelement.

Beispielsweise dreht der Befehl `\rotatebox{15}{\_top\_}` das angegebene Argument um 15°. Gedreht wird gegen den Uhrzeigersinn um den linken Schnittpunkt der das Argument umgebenden gedachten Box mit der Grundlinie: `\_top\_`

Durch Angabe von Optionen kann der Drehpunkt abweichend festgelegt werden. Eine weitere Option erlaubt die Festlegung des Drehwinkels in anderen Einheiten als in den voreingestellten Grad gegen den Uhrzeigersinn.

Optionen:

origin=<Drehpunkt>

Spezielle Drehpunkte können durch eine Kennung, bestehend aus ein oder zwei Buchstaben, vorgegeben werden. Die Buchstaben haben dabei die folgende Bedeutung:

- c** steht für den Mittelpunkt der Box, wenn ein zweiter Buchstabe fehlt, oder bei Vorgabe des Drehpunkts in einer Richtung für die Mitte der Box in der zweiten Richtung.
- l** Drehpunkt am linken Rand und
- r** am rechten Rand
- t** Drehpunkt am oberen oder
- b** am unteren Rand der Box und
- B** Drehpunkt auf der Grundlinie.

x=<Länge> | y=<Länge>

Diese Optionen verschieben den Drehpunkt horizontal (x) bzw. vertikal (y) um die angegebenen Länge gegenüber dem Referenzpunkt der Box. Diese Angaben erlauben eine beliebige Positionierung des Drehpunkts *innerhalb und außerhalb* des Textelements.

units=<Zahl>

Durch diese Angabe können die Einheiten für die Angabe des Drehwinkels festgelegt werden. *<Zahl>* ist dabei diejenige Zahl, die einem vollen Kreis beim Durchlaufen gegen den Uhrzeigersinn entspricht. Ein negativer Wert bewirkt daher eine Drehung im Uhrzeigersinn. Voreingestellt ist der Wert 360. Für eine Angabe des Drehwinkels im Uhrzeigersinn in Grad ist also die Option `units=-360`

anzugeben. Angaben im Bogenmaß (gegen den Uhrzeigersinn) sind mit der Option `units=6.283185` möglich.

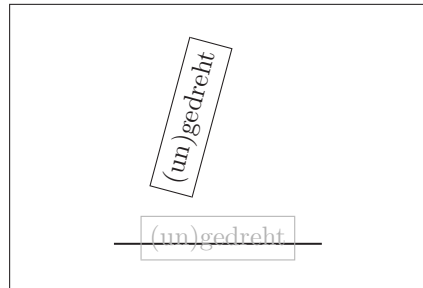
Abbildung 10.2 zeigt als Beispiel ein um 30° gedrehtes Textelement bestehend aus einer leeren `\fbox`, über die mit höherer Strichstärke die Grundlinie und in verringerter Strichstärke die horizontalen und vertikalen Mittellinien der Box gezeichnet werden. Die Abbildung zeigt die Möglichkeiten zur Angabe des Drehpunkts mit Hilfe von `origin`. Bei Angabe von zwei Buchstaben zur Festlegung des Drehpunkts spielt die Reihenfolge keine Rolle. Üblicherweise wird allerdings über den ersten Buchstaben die horizontale und über den zweiten die vertikale Position festgelegt, also typischerweise `lB` (Referenzpunkt) oder `rt` (rechte obere Ecke). Die ungedrehte Box mit ihren Hilfslinien ist jeweils grau wiedergegeben. Darüber ist in normaler Schriftfarbe die gedrehte Box gesetzt. Sie ist horizontal so verschoben, dass die Drehpunkte der gedrehten und ungedrehten Box zusammenfallen. Ohne diese Verschiebung würde das gedrehte Textelement linksbündig über das ungedrehte gesetzt.

Für die allgemeine Festlegung des Drehpunkts mit `x` und `y` wird ein Beispiel gegeben. Auf eine horizontale Verschiebung wie in Abbildung 10.2 wird hier verzichtet. Die Grundlinie ist durch einen Strich markiert. Die ungedrehte Box ist wie in Abbildung 10.2 grau wiedergegeben. Der Drehpunkt wurde hier 7 mm *vor* den zu drehenden Text auf die Grundlinie gelegt. Dadurch ‚schwebt‘ die gedrehte Box über der Grundlinie.

```

6  \usepackage[dvips]{graphicx,color}
7  \definecolor{grau}{gray}{.7}
8  \begin{document}
9  \rule{1em}{.75pt}%
10 \rule{\Boxwd}{.75pt}%
11 \textcolor{grau}{%
12   \hspace{-\Boxwd}%
13   \fbox{(un)gedreht}}}%
14 \rule{1em}{.75pt}%
15 \hspace{-1em}\hspace%
16   {-\Boxwd}
17 \rotatebox[x=-7mm,y=0mm]
18   {75}{\fbox{(un)gedreht}}
19 \end{document}

```



[Buch1/Grafik/RotXY.tex]

10.3.2 Skalieren von Ausgabeelementen

Auch das Skalieren in horizontaler und vertikaler Richtung, mit unterschiedlichen Skalierungsfaktoren, ist mit den Paketen möglich. Neben der Vorgabe der Skalierungsfaktoren kann man die gewünschte

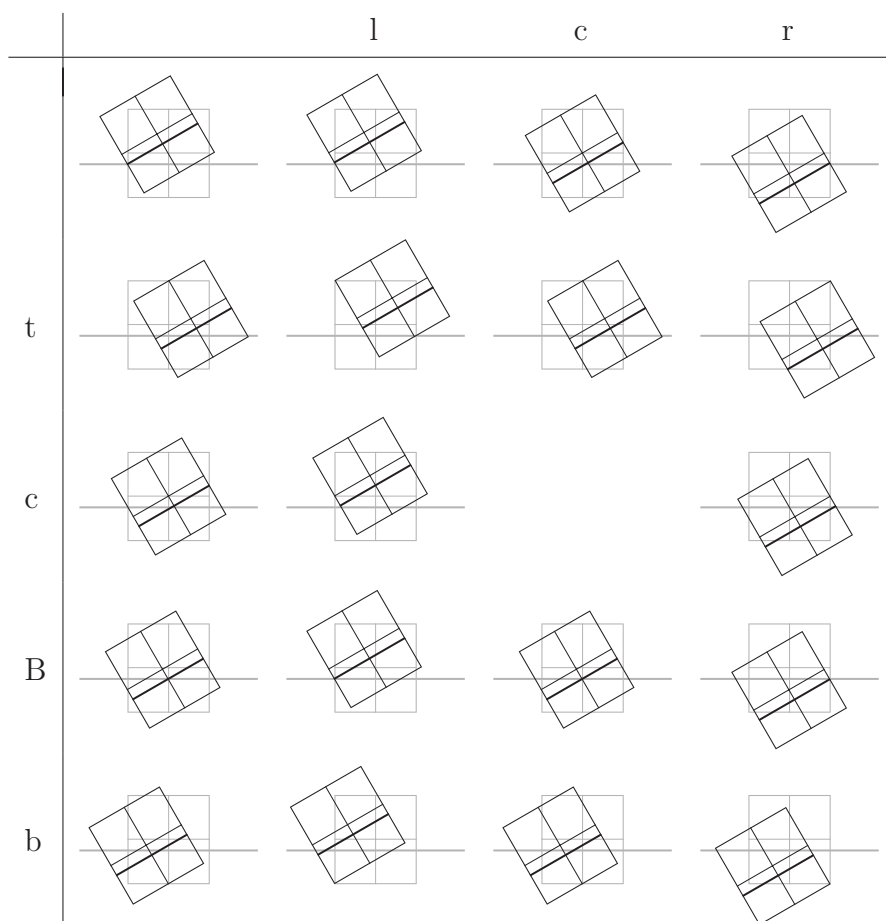


Abbildung 10.2: Drehen von Textelementen bei Vorgabe des Drehpunkts mit Hilfe von `origin`

Größe des skalierten Ausgabeelements vorgeben. Bedingung ist wieder, dass das zu skalierende Ausgabeelement in eine Box gepackt werden kann.

Zum Skalieren eines Ausgabeelements `<Text>` durch Vorgabe eines Skalierungsfaktors oder getrennter Faktoren zur horizontalen und vertikalen Skalierung dient der folgende Befehl:

```
\scalebox{<H-Faktor>}[<V-Faktor>]{<Text>}
```

Bei fehlendem optionalem Argument *<V-Faktor>* wird das Argument *<Text>* horizontal und vertikal mit dem Faktor *<H-Faktor>* skaliert. Andernfalls wird er horizontal mit *<H-Faktor>* und vertikal mit *<V-Faktor>* skaliert.

Der Befehl `\scalebox{.75}{verkleinert}` setzt also das angegebene Wort um ein Viertel (25 %) verkleinert in den Text: verkleinert, während der Befehl `\scalebox{1.5}[1]{gespreizt}` das Wort horizontal um 50% verbreitert, es aber in seiner Höhe unverändert lässt: gespreizt.

Zusätzlich steht der Befehl

```
\reflectbox{<Text>}
```

als Kurzform des Befehls `\scalebox{-1}[1]{<Text>}` zur Verfügung.

Zur Ausgabe eines Elements in einer vorgegebenen Größe dient der Befehl

```
\resizebox*{<Breite>}{<Höhe>}{<Text>}
```

Das Ausgabeelement *<Text>* wird durch den Befehl so skaliert, dass es *<Breite>* und *<Höhe>* erhält. Soll das Ausgabeelement nicht verzerrt werden, kann man nur einen der Werte festlegen; durch Angabe des Zeichens ! für den fehlenden Wert wird der sich aus dem anderen Argument ergebenden Skalierungsfaktor übernommen. In den Angaben für Höhe und Breite können auch `\height` für die natürliche Höhe (oberhalb der Grundlinie), `\depth` für die Unterlänge, `\totalheight` für die Gesamthöhe (`\height+\depth`) und `\width` für die Breite des Ausgabeelements verwendet werden.

Das Argument *<Höhe>* bezieht sich in der Grundform des Befehls (also ohne den Stern *) auf die Höhe des skalierten Ausgabeelements (oberhalb der Grundlinie). Ein Beispiel verdeutlicht das:

```

7  \usepackage{graphicx}
8  \begin{document}
9  \resizebox{2.5cm}{\height}%
10   {gespreizt}
11   setzt das Wort in einer
12   Breite von 2,5\,cm und
13   seiner natürlichen Höhe.
14 \end{document}
```

gespreizt setzt das Wort in einer Breite von 2,5 cm und seiner natürlichen Höhe.

[Buch1/Grafik/spreiz.tex]

In der Stern-Form bezieht sich die Höhenangabe statt dessen auf die Gesamthöhe.

```

9  Das gleiche Ergebnis
10 erzielt\\
11 \resizebox*{2,5cm}%
12   {\totalheight}%
13   {gespreizt}.
14
15 Unverzerrte Verkleinerung auf
16 die H"ohe der Kleinbuchstaben
17 \resizebox{!}{1ex}{verkleinert}.

```

Das gleiche Ergebnis erzielt
gespreizt.
 Unverzerrte Verkleinerung auf die
 Höhe der Kleinbuchstaben verkleinert.

[Buch1/Grafik/spreiz2.tex]

10.3.3 Einbinden von Bildern

Zum Einbinden von Bildern wird in beiden Paketen ein Befehl verwendet, dessen Form unabhängig ist vom Typ des eingebundenen Bilds. Dabei wird vorausgesetzt, dass der Typ des Bilds an der Erweiterung des Dateinamens zu erkennen ist. Ob ein bestimmter Bildtyp eingebunden werden kann, hängt allerdings vom verwendeten Ausgabegerät und damit vom Treiber ab. Die Bildtypen, die von den verschiedenen Treibern akzeptiert werden, und die zugehörigen Erweiterungen der Dateinamen sind in Tabelle 10.4 auf S. 321 aufgeführt.

Bei abweichenden Dateitypen können unbekannte Erweiterungen leicht hinzugefügt werden. Auch die Erweiterung der einem Treiber bekannten Bildtypen ist möglich, wenn es Befehle zur Wandlung in einen bekannten Bildtyp gibt. Erkennt der Treiber eine Erweiterung des Dateinamens nicht, wird ein treiberspezifisch voreingestellter Bildtyp angenommen. Die PostScript-Treiber nehmen beispielsweise an, dass es sich um ein PostScript-Bild des Typs `eps` handelt. Der voreingestellte Bildtyp kann bei Bedarf in der Konfigurationsdatei des Treibers angepasst werden.

Je nach Bildtyp und Ausgabegerät bieten die Treiber allerdings unterschiedliche Möglichkeiten, das Bild beim Einbinden zu beeinflussen. Tabelle 10.4 gibt einen Überblick über diese Fähigkeiten der Treiber. Dazu gehört beispielsweise das Skalieren und Drehen der Bilder oder das Herausschneiden eines Bildausschnitts durch Angabe von Optionen beim Einbinden der Bilder.

Die beiden Pakete `graphics` und `graphicx` verwenden im einfachsten Fall, d. h. wenn keine Option angegeben werden, den gleichen Befehl zum Einbinden von Bildern. In der vollständigen Form der Befehle gibt es jedoch Unterschiede:

```

graphics: \includegraphics*[<LUx,LUy>][<RUx,RUy>][<Datei>}
graphicx: \includegraphics*[<Option(en)>][<Datei>}

```

`<Datei>` ist der Name der Datei, die die Bilddaten enthält. Der Dateiname kann ohne Erweiterung angegeben werden; in diesem Fall wird die treiberspezifisch voreingestellte Erweiterung angehängt.

In der `*`-Form werden Bildteile außerhalb des im Befehl oder im Bild angegebenen Bildbereichs abgeschnitten (Clipping). Andernfalls werden derartige Bildteile bei der Reservierung von Platz für das Bild ignoriert, aber ausgegeben; sie können also andere Ausgabeelemente überlagern.

Fehlen Angaben zur Größe des eingebundenen Bilds, wird die Größe aus der Bilddatei bestimmt – bei Rasterbildern aus der Anzahl der horizontalen und vertikalen Pixel und bei PostScript-Bildern aus der expliziten Größenangabe in der PostScript-Datei.

Da die optionalen Teile der beiden Befehlsformen sich grundsätzlich unterscheiden, werden sie getrennt behandelt.

Paket graphics

Hier hat der Befehl zwei optionale Argumente. Das erste optionale Argument `<LUx, LUy>` legt die Koordinaten der linken unteren, das zweite `<ROx, ROy>` die der rechten oberen Ecke des Bilds fest. Die vier Koordinatenangaben `<LUx>`, `<LUy>`, `<ROx>` und `<ROy>` sind Längenmaße. Die Angabe der Längeneinheit `bp` kann allerdings fehlen und wird dann automatisch ergänzt (in dieser Form finden sich Größenangaben in den PostScript-Dateien). Die beiden Angaben `1in, 1in` und `72, 72` sind daher gleichwertig. Ist nur ein optionales Argument angegeben, legt es die rechte obere Ecke fest; für die linke untere Ecke werden die Koordinaten `0, 0` ergänzt. Fehlen beide optionalen Argumente, wird die Größe – wie oben erwähnt – aus der Bilddatei bestimmt.

Paket graphicx

Bei diesem Paket werden alle über den Dateinamen hinausgehenden Angaben in einem einzigen optionalen Argument in Form einer Optionsliste vorgenommen. Die `*`-Form ist hier nur aus Gründen der Kompatibilität vorgesehen, da das Abschneiden oder Anzeigen von Bildteilen außerhalb des gültigen Bildbereichs auch durch eine Option gesteuert werden kann. Die einzelnen Optionen in der Liste werden durch Kommas getrennt. Alternativ kann auch die Form des Befehls beim Paket `graphics` verwendet werden.

In den `<Option(en)>` sind eine Vielzahl von Angaben möglich. In der folgenden Liste der verfügbaren Optionen werden logisch zusammengehörende Angaben gemeinsam behandelt.

Festlegung der Größe oder des Ausschnitts des Originalbildes:

bb=<LUx> <LUy> <ROx> <ROy>

Vorgabe des Bildbereichs des eingebundenen Bilds durch die Koordinaten der linken unteren und der rechten oberen Ecke *Bounding Box*. Die Koordinaten werden in der gleichen Weise angegeben wie im Fall des Pakets `graphics`. Die Wiedergabe von Bildteilen außerhalb des Bildbereichs hängt von den *Clipping*-Einstellungen ab.

bbllx=<LUx>, bblly=<LUy>, bburx=<ROx>, bbury=<ROy>

Alternative Form der Vorgabe eines Bildbereichs. Die Angabe aller vier Einzeloptionen in der gezeigten Form ist gleichbedeutend mit der oben angegebenen Form der Option `bb`. Fehlende Angaben werden aus den Angaben der *BoundingBox* ergänzt, die dann zur Verfügung stehen oder als Option `bb` angegeben werden muss.

natwidth=<Breite>, natheight=<Höhe>

Eine weitere Alternative zur Vorgabe eines Bildbereichs. Die Angabe beider Einzeloptionen in der gezeigten Form ist gleichbedeutend mit der Option `bb=0 0 <Breite> <Höhe>`. Wird nur die Breite oder nur die Höhe festgelegt, werden die fehlenden Angaben zu den Koordinaten aus der *BoundingBox* ergänzt, die in diesem Fall zur Verfügung stehen muss. Gegebenenfalls kann eine fehlende *BoundingBox* als Option `bb` angegeben werden.

hiresbb | hiresbb=true | hiresbb=false

Die beiden ersten Formen sind gleichbedeutend und bewirken, dass in der eingebundenen Bilddatei nach einer Größenangabe der Form

`%%HiResBoundingBox: <LUx> <LUy> <ROx> <ROy>`

gesucht wird, die eine Angabe der Koordinaten der Ecken mit Nachkommastellen erlaubt. Die Option `hiresbb=false` schaltet auf die Suche nach der Standardform `%%BoundingBox:` um und kann verwendet werden, um eine globale Option `hiresbb` beim Laden des Pakets `graphicx` zu überschreiben.

viewport=<LUx> <LUy> <ROx> <ROy>

Wählt ähnlich der bereits beschriebenen Option `bb` einen Bildbereich aus. Die Koordinaten werden in der gleichen Form wie bei `bb` angegeben. Im Unterschied zu `bb` sind die Angaben jedoch *relativ* zur linken unteren Bildecke, wie sie sich aus der Bilddatei ergibt. Die linke untere Bildecke hat daher die Koordinaten (0, 0), und die Angabe `viewport=0 0 5cm 3cm` wählt einen 5 cm breiten und

3 cm hohen Bildausschnitt in der linken unteren Bildecke aus dem Bild aus.

trim=<Links> <Unten> <Rechts> <Oben>

Bei den Angaben <Links>, <Unten>, <Rechts> und <Oben> handelt es sich um Längenmaße, bei denen die Längeneinheit bp allerdings fehlen darf und dann automatisch ergänzt wird (wie bei der Länge <LUx>). Die Angaben legen die Breite eines entsprechenden Randes fest, der abgeschnitten wird.

Drehen und Skalieren des Bilds:

angle=<Winkel>

Dreht das Bild um den angegebenen Winkel. <Winkel> hat die gleiche Bedeutung wie das gleichnamige Argument beim Befehl `\rotatebox` auf S. 330.

origin=<Drehpunkt>

Legt den Drehpunkt fest. <Drehpunkt> hat die gleiche Bedeutung wie die gleichnamige Option beim Befehl `\rotatebox` auf S. 330.

width=<Breite>

Skalieren des eingebundenen Bilds auf die angegebene <Breite>. Diese ist ein Längenmaß, beispielsweise `\textwidth`. Wird die Höhe nicht ebenfalls vorgegeben, erfolgt eine vertikale Skalierung mit dem gleichen Skalierungsfaktor; andernfalls wird das Bild in der Regel verzerrt.

height=<Höhe>

Skalieren des eingebundenen Bilds auf die angegebene Höhe über der Grundlinie (häufig – aber nicht notwendigerweise – identisch mit dem unteren Bildrand). <Höhe> ist ein Längenmaß. Fehlt eine Vorgabe für die Breite, wird horizontal mit dem gleichen Skalierungsfaktor skaliert; andernfalls wird das Bild in der Regel verzerrt.

totalheight=<Gesamthöhe>

Die Option `height` bezieht sich lediglich auf die Höhe des Bilds oberhalb der Grundlinie, entsprechend der $\text{\LaTeX 2}_{\epsilon}$ -Länge `\height`. Solange das Bild nicht gedreht wird, steht das Bild auf der Grundlinie und hat daher keine Unterlänge ($\text{\LaTeX 2}_{\epsilon}$ -Länge `\depth`). Bei Drehung eines 5 cm×3 cm großen Bilds um -90° (um seine linke untere Ecke) erhält man ein Bild der Größe 3 cm×5 cm, das eine Unterlänge (`\depth`) von 5 cm hat, während `\height` den Wert 0 cm hat und damit die Option `height` sogar ihren Sinn verliert. In weniger krassen Fällen bewirkt die Drehung nur, dass die Höhe des Bilds oberhalb der Grundlinie und die Gesamthöhe nicht mehr übereinstimmen. In solchen Fällen kann man die Höhe des

Bilds mit der Option `totalheight` festlegen, die der L^AT_EX-Länge `\totalheight = \height + \depth` und damit der Gesamthöhe – der Summe aus der Höhe oberhalb der Grundlinie und der Unterlänge – des Bilds entspricht.

keepaspectratio

keepaspectratio=true | keepaspectratio=false

Die ersten beiden Formen bewirken, dass sich beim Skalieren des Bilds das Verhältnis zwischen Breite und Höhe nicht verändert. In diesem Fall wird das Bild so skaliert, dass keine der beiden Vorgaben überschritten wird (unabhängig davon, welche der beiden Optionen `height` und `totalheight` verwendet wurde).

scale=<Faktor>

Vorgabe eines Skalierungsfaktors für das Bild.

Clipping und Wiedergabe des Bilds:

clip | clip=true | clip=false

Bei Angabe der ersten beiden Formen der Option werden Bildteile außerhalb des Bildbereichs (explizit festgelegt oder aus der Bilddatei übernommen) abgeschnitten. Die dritte Form verlangt hingegen ausdrücklich, dass derartige Teile des Bilds gedruckt werden, obwohl sie bei der Reservierung des Platzes für das Bild nicht berücksichtigt werden und damit gegebenenfalls andere Ausgabeelemente überschreiben.

draft | draft=true | draft=false

Die ersten beiden Formen wählen den Draft-Modus, in dem die Ausgabe des Bilds unterdrückt und statt dessen nur ein Rahmen in Bildgröße ausgegeben wird, der den Namen der Bilddatei enthält. Die dritte Form ist nur von Interesse, wenn global (im `\includegraphics`-Befehl) durch Angabe der Option `draft` der Draft-Modus eingeschaltet wurde und das aktuelle Bild trotzdem ausgegeben werden soll.

Name und Behandlung des Bilds:

type=<Grafiktyp>

Festlegung des Grafiktyps des aktuellen Bilds wie auf S. 343 beim Befehl `\DeclareGraphicsRule`.

ext=<Extension>

Festlegung der Erweiterung des Dateinamens für das aktuelle Bild wie für den Befehl `\DeclareGraphicsRule` auf S. 343 beschrieben. Anm.: Diese Option sollte nur in Verbindung mit der Option `type` verwendet werden!

read=<Extension>

Festlegung der Erweiterung des Dateinamens einer dem aktuellen Bild zugeordneten Datei, der Angaben über die Größe des Bilds entnommen werden, wie es beim Befehl `\DeclareGraphicsRule` auf S. 343 beschrieben ist.

Anm.: Diese Option sollte nur in Verbindung mit der Option `type` verwendet werden!

command=<cmd>

Festlegung eines Befehls, der dem Treiber zum Lesen der Bilddatei übergeben wird (meist leer). Eine genauere Beschreibung finden Sie beim Befehl `\DeclareGraphicsRule` auf S. 343.

Anm.: Diese Option sollte nur in Verbindung mit der Option `type` verwendet werden!

Bei Verwendung dieser Gruppe von Optionen sollte der Name der Bilddatei *ohne die Erweiterung des Dateinamens* angegeben werden.

Hinweis:

Normalerweise spielt die Reihenfolge, in der Optionen angegeben werden, keine Rolle. Das trifft nicht zu, wenn das Bild sowohl gedreht als auch an eine vorgegebene Größe angepasst werden soll.

Wird zuerst eine der Optionen `width`, `height` und `totalheight` angegeben und anschließend das mit Hilfe der Option `angle` gedreht, wird das Bild auch zuerst in der Größe angepasst und erst anschließend gedreht. Durch eine nachfolgende Drehung ändert sich in der Regel der zur Wiedergabe benötigte Platz, während die Maße des Bildbereichs exakt den Vorgaben entsprechen.

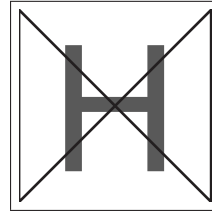
Wird das Bild hingegen zunächst gedreht und anschließend eine Größe vorgegeben, wird der Platzbedarf für das gedrehte Bild festgelegt. In diesem Fall weichen die Maße des Bildbereichs in der Regel von der vorgegebenen Breite oder Höhe ab.

In den Beispielen zum Einbinden von Bildern wird das folgende kleine PostScript-Bild verwendet:

```

%!
%%BoundingBox: 100 100 172 172
112 111.5 moveto
/Helvetica findfont
66 scalefont
setfont
.35 setgray
(H) show
0 setgray
100 100 moveto
72 72 rlineto
0 72 neg rlineto
72 neg 72 rlineto
closepath
stroke
showpage

```



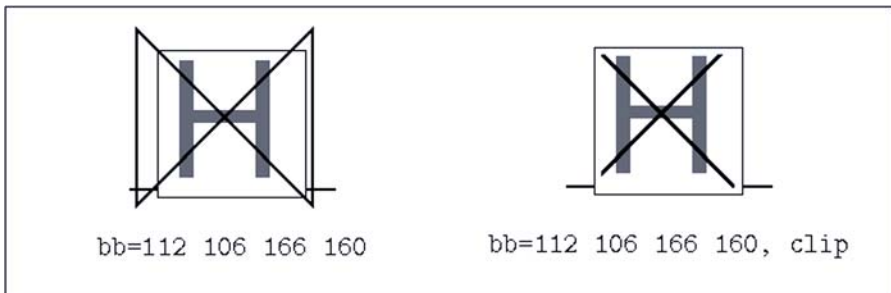
Zum Einbinden wird jeweils die Eingabe

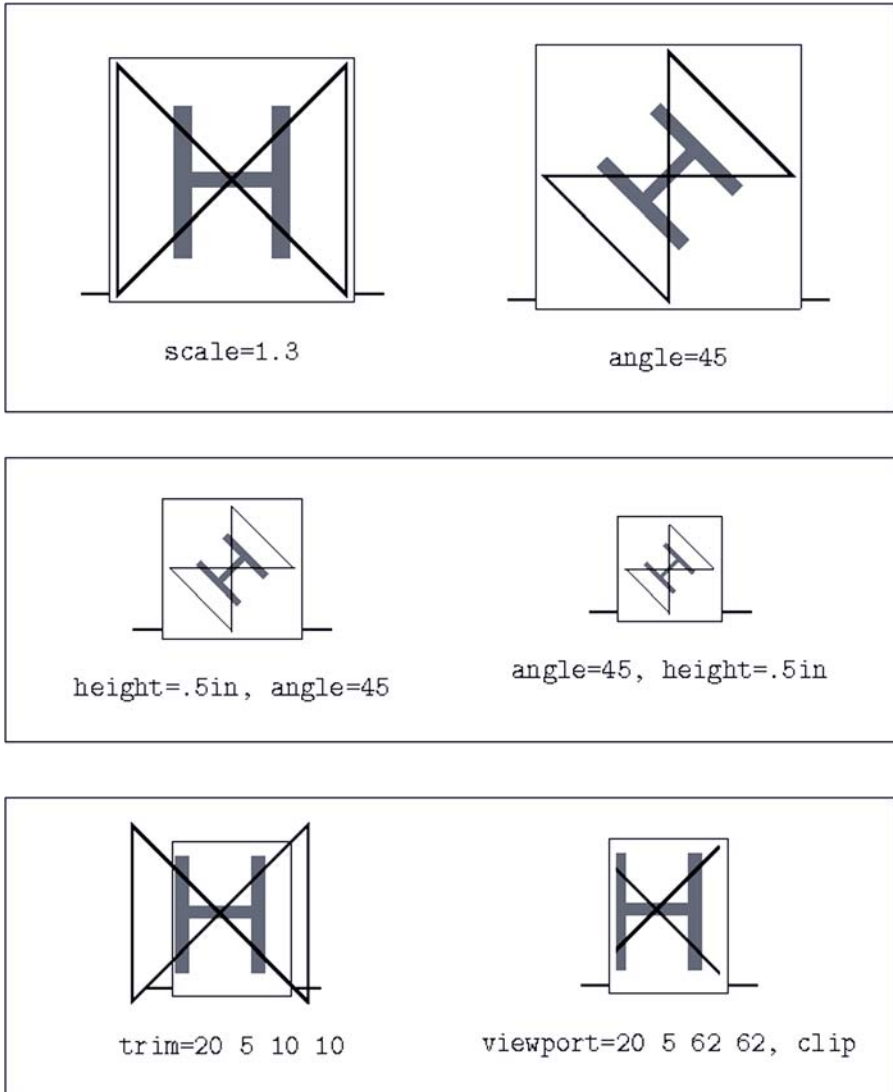
```

\rule{1em}{1pt}%
\fbbox{\includegraphics[<Optionen>]{incxmpl.ps}}%
\rule{1em}{1pt}%

```

verwendet. Die beiden `\rule`-Befehl markieren jeweils die Grundlinie. `\fbbox` wird verwendet, um die von $\text{\LaTeX}$ berücksichtigten Ausmaße des Bilds zu zeigen. Der Abstand zwischen dem Rand des Bilds und dem Rahmen beträgt `\fbboxsep` (hier `3.0pt = 1.05438 mm`). Bei jedem Beispiel werden die verwendeten `<Optionen>` angegeben. Das Bild oben wurde ohne `\rule` und ohne Angabe von Optionen gesetzt.



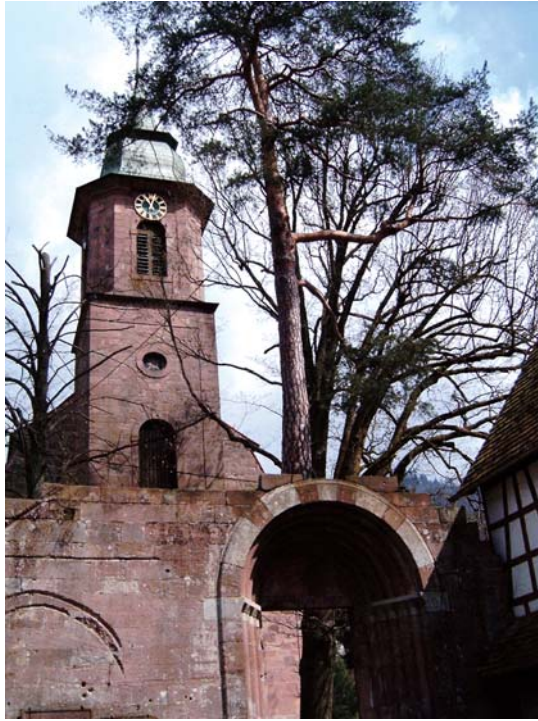


Als abschließendes Beispiel für das Einbinden von Bildern wird ein Farbfoto im png-Format eingebunden. Das Foto wurde mit einer HP-635 Digitalkamera im JPEG-Format aufgenommen und hat eine Auflösung von 1200×1600 Punkten.

```

4 \usepackage[dvipdfm]{graphicx,color}
5 \begin{document}
6 \begin{center}
7 \includegraphics[width=2.8in]{../.. /Bilder/Graphics/foto.png}
8 \end{center}
9 \end{document}

```



[Buch1/Grafik/foto.tex]

10.3.4 Vereinbarungen beim Einbinden von Bildern

Neben den Befehlen zum Einbinden von Bildern ist es zweckmäßig, den Suchpfad für die einzubindenden Bilder vorgeben zu können, damit er nicht jedesmal wiederholt werden muss. Darüberhinaus stehen Befehle zur Verfügung, um dem Treiber zusätzliche Erweiterungen von Dateinamen bekannt zu machen oder das Einbinden von Bildern in anderen Bildformaten durch Festlegung von Systembefehlen zur Umwandlung in dem Treiber bekannte Bildformate zu ermöglichen.

Zur Vorgabe des Suchpfads für eingebundene Bilder kann man den Befehl

```
\graphicspath{<Verzeichnisliste>}
```

verwenden. Die Verzeichnisliste besteht aus einem oder mehreren Namen von Verzeichnissen oder Unterverzeichnissen, die *einzeln* in geschweifte Klammern einzuschließen sind. Um Bilder in den Unterverzeichnissen pdf und tiff des aktuellen Verzeichnisses zu suchen, könnte unter Unix oder Windows etwa der Befehl

```
\graphicspath{{pdf/}{tiff/}}
```

verwendet werden. Bei MacIntosh-Rechnern würde der Befehl

```
\graphicspath{{:pdf:}{:tiff:}}
```

lauten. Ohne diesen Befehl würden Bilder nur im normalen Suchpfad gesucht, also dort wo auch Eingabedateien gesucht werden.

Der nächste Befehl legt fest, welche Dateinamenserweiterungen $\text{\LaTeX}$ beim Einbinden einer Bilddatei ausprobieren soll, wenn der Dateiname ohne Erweiterung angegeben ist:

```
\DeclareGraphicsExtensions{<Erweiterungsliste>}
```

<Erweiterungsliste> ist dabei eine durch Kommas getrennte Liste von möglichen Erweiterungen des Dateinamens, falls nicht explizit eine Erweiterung im `\includegraphics`-Befehl angegeben ist. Wird beim Einbinden einer Bilddatei keine Datei des angegebenen Namens gefunden, wird aus dem angegebenen Namen und der ersten Erweiterung ein Dateiname gebildet und nach einer Datei mit dem so gebildeten Namen gesucht. Ist diese Suche erfolgreich, verhält sich der `\includegraphics`-Befehl so, als wäre der vollständige Name angegeben. Andernfalls wird der Versuch mit der nächsten angegebenen Erweiterung wiederholt. Die Suche wird erst aufgegeben, wenn alle Erweiterungen durchprobiert worden sind.

Ist der Name der einzubindenden Bilddatei im `\includegraphics`-Befehl nicht vollständig angegeben, muss die Bilddatei existieren, wenn $\text{\LaTeX}$ ausgeführt wird, da – wie gerade beschrieben – nach einer Datei des durch Anhängen der Erweiterung gebildeten Namens gesucht wird. Ist hingegen der vollständige Name angegeben, braucht die Bilddatei nicht zu existieren, wenn die Bildgröße als Option im `\includegraphics`-Befehl vorgegeben ist oder sich aus der von $\text{\LaTeX}$ zu lesenden Datei – der sogenannten *read*-Datei – ergibt, die dann natürlich vorhanden sein muss.

Der letzte Befehl in den beiden Grafikpaketen erlaubt es, den Umgang des Treibers mit Bildern festzulegen und dabei gegebenenfalls festzulegen, wie Bilder in einem dem Treiber unbekannten Bildformat in ein bekanntes Bildformat umgewandelt werden sollen:

```
\DeclareGraphicsRule{<Ext>}{<Typ>}{<RD-Datei>}{<Befehl>}
```

Durch Verwendung mehrerer `\DeclareGraphicsRule`-Befehle können Regeln für den Umgang von $\LaTeX$ mit verschiedenen Bildformaten vereinbart werden.

`<Ext>` ist die Erweiterung des Dateinamens, für die das Verhalten festgelegt wird. Sollen die Regeln für *sämtliche* unbekannten Erweiterungen vorgegeben werden, kann als Sonderform für `<Ext>` der Wert `*` angegeben werden; beispielsweise legt der Befehl `\DeclareGraphicsRule{*}{eps}{*}{}{}` fest, dass alle Bilder mit einer unbekannten Erweiterung ihres Dateinamens wie Bilder des Typs Encapsulated PostScript behandelt werden sollen und dass zur Bestimmung ihrer Größe in der Bilddatei selbst nach der Kommentarzeile `%%BoundingBox:` gesucht werden soll.

`<Typ>` legt den Typ des Bilds fest. Alle Bilder des gleichen Typs werden vom Treiber mit demselben internen Befehl – der im Treiber festgelegt sein muss – eingelesen. So kann man zum Beispiel alle Dateien mit den Erweiterungen `.ps`, `.ps.gz`, `.eps` und `.eps.gz` dem gemeinsamen Typ `eps` zuordnen.

Das Argument `<RD-Datei>` legt die Erweiterung derjenigen Datei fest, die von $\LaTeX$ gelesen werden soll, um die Größe des Bilds herauszufinden. Um die Größe der Bilddatei selbst zu entnehmen, kann die Erweiterung der Bilddatei `<ext>` zu wiederholt oder `*` als Abkürzung angegeben werden. Enthält dieses Argument *keine Angaben* und besteht daher nur aus einem Paar geschweifter Klammern, so versucht der Treiber nicht, die Größe selbst zu bestimmen (durch Lesen einer Datei); statt dessen *muss* die Größe als Argument im `\includegraphics`-Befehl angegeben sein.

10.4 Weitere Pakete

Teil der Grafiksammlung sind auch einige weitere Pakete, die die eigentlichen Grafikpakete unterstützen oder Kompatibilität mit älteren Paketen herstellen. `lscap` ist ein Paket, um den Inhalt einer Seite (ohne Kopf- und Fußzeilen) um 90° zu drehen. Mit dem Paket `epsfig` lassen sich Dokumente formatieren, die mit dem veralteten Paket `psfig` geschrieben wurden. Das Paket `pstcol` dient dazu, eine Inkompatibilität zwischen den Paketen `color` und `pstricks` zu lösen. Lediglich Hilfsfunktionen haben die Pakete `keyval` und `trig`.

10.4.1 Paket `lscape`

Dieses Paket ist das einzige Zusatzpaket, das den Funktionsumfang der Grafikpakete erweitert. Es wird geladen mit

```
\usepackage[<Option(en)>]{lscape}
```

Es können alle auf S. 328 für das Paket beschriebenen Optionen angegeben werden. Eine Beschreibung der Optionen an dieser Stelle ist daher nicht erforderlich. Die Angabe eines Treibers kann nur entfallen, wenn in der Konfigurationsdatei `graphics.cfg` bereits der richtige Treiber festgelegt wird.

Das Paket stellt eine Umgebung `landscape` zur Verfügung, die eine Seite im Querformat setzt:

```
\begin{landscape}
...
\end{landscape}
```

Kopf- und Fußzeilen sind von der Drehung um 90° gegen den Uhrzeigersinn ausgenommen. Die Umgebung enthält den Inhalt genau einer Seite. Ein Seitenumbruch findet nicht statt. Explizite Seitenumbrüche im Dokument – beispielsweise mit dem Befehl `\pagebreak` – sind wirkungslos.

Die Umgebung ist insbesondere geeignet, um Tabellen zu setzen, die breiter sind als die Textbreite, aber gedreht noch auf eine Seite passen. Auch ganzseitige Abbildungen im Querformat mit Bildunterschrift lassen sich gut mit dieser Umgebung setzen.

10.4.2 Paket `epsfig`

Bereits vor der Weiterentwicklung von $\text{\LaTeX}$ 2.09 zu $\text{\LaTeX}$ 2_ε und damit vor der Entstehung des hier beschriebenen Grafikpakets entstanden sogenannte ‚Styles‘ (Vorgänger von Paketen) zum Einbinden von PostScript-Bildern. Einer der leistungsfähigsten ‚Styles‘ war `psfig`, der später auch in ein Paket umgewandelt wurde. Mit der Entwicklung der Pakete `graphics` und `graphicx` zu einem Standard nahm die Nutzung von `psfig` ab. Da die Standard-Grafikpakete mehr Möglichkeiten als `psfig` bieten, wurde schließlich das Paket `epsfig` entwickelt, das den Befehl

```
\psfig{file=<Datei>}[<Weitere Option(en)>]
```

mit Hilfe des `\includegraphics`-Befehls emuliert. Damit genügt es zum Formatieren bestehender Dokumente, die das Paket `psfig` verwenden, den Paketnamen `psfig` durch `epsfig` zu ersetzen. Bei neuen Dokumenten empfiehlt es sich aber, das Paket `graphics` oder `graphicx` mit den Befehl `\includegraphics` einzusetzen.

10.4.3 Paket `pstcol`

Das Paket `pstricks` von Timothy Van Zandt ist ein sehr leistungsfähiges Interface zwischen $\text{\TeX}$ und PostScript. Es bietet vielfältige Möglichkeiten, PostScript-Bilder in einem $\text{\LaTeX}$ -Dokument zu erstellen. Dieses umfangreiche Paket wird in einem späteren Band näher beleuchtet.

Leider ist die Unterstützung von Farben durch `pstricks` (bisher) nicht voll kompatibel mit dem Paket `color`. Diese Inkompatibilität beseitigt das Paket `pstcol`, wenn es statt `pstricks` geladen wird. `pstcol` lädt das Paket `pstricks` und ändert einige der dort definierten Befehl so ab, i dass `pstricks` alle mit Befehlen aus dem Paket `color` definierten Farben verwenden kann und umgekehrt.

10.4.4 Paket `keyval`

Dieses Hilfspaket von David Carlisle ermöglicht es anderen Paketen, durch Kommas getrennte Listen von Optionen in der Form `<Optname> [= <Wert>]` zu erkennen und zu verarbeiten. Pakete, die diese Form von Optionen unterstützen, laden das Paket in der Regel automatisch, falls es nicht bereits geladen ist. Da `keyval` keine Befehle für Anwender enthält, wird es üblicherweise nicht direkt geladen.

Das Paket ist ausdrücklich dazu gedacht, auch von anderen Paketen verwendet zu werden. Einzelheiten für Entwickler von Paketen enthält die Paketdokumentation.

10.4.5 Paket `trig`

Auch dieses Paket ist nicht dazu gedacht, vom Anwender direkt aufgerufen zu werden. Seine Aufgabe ist es, Werte der trigonometrischen Funktionen Sinus, Cosinus und Tangens zu berechnen, die beispielsweise beim Drehen von Bildern oder Boxen benötigt werden, um Breite und Höhe des gedrehten Objekts zu ermitteln.

PostScript-Zeichensätze

Die Sammlung `psnfss` fasst eine Reihe von Einzelpaketen zusammen, mit denen in $\text{\LaTeX}$ -Dokumenten PostScript-Schriften (engl. *fonts*) statt der $\text{\TeX}$ -typischen CM- oder EC-Fonts verwendet werden können. Diese Sammlung unterstützt die 35 Standard-PostScript-Fonts, die auf jedem PostScript-Drucker zur Verfügung stehen. Zusätzlich wird der frei verfügbare PostScript-Font ‚Bitstream Charter‘ unterstützt. PostScript ist eine Entwicklung der Firma Adobe Inc.

Damit die Zeichensätze bei der Ausgabe angezeigt werden können, muss man ein PostScript fähiges Gerät verwenden, in aller Regel ein Drucker. Hat man keinen PostScript-Drucker zur Verfügung, so kann mit der unter der GNU-Lizenz kostenlos verfügbaren Software GhostScript gearbeitet werden, die PostScript für Nicht-PostScript-Drucker emuliert. Zur Ausgabe des Formats PostScript an einem Bildschirm gibt es die Software GhostScript. In der Regel haben PostScript-Dateien die Dateinamenserweiterung `.ps` oder `.eps`.

In diesem Kapitel werden die Namen der Zeichensätze (Fonts) in einfache Anführungszeichen gesetzt. Dies dient zum einen der Lesbarkeit, da die Namen teilweise aus mehreren Worten bestehen, zum anderen zur Vermeidung von Verwechslungen beispielsweise zwischen dem Begriff Symbol und dem ‚Symbol‘-Font.

Tabelle 11.1 gibt eine Übersicht über die aktuellen Pakete zur Verwendung dieser Fonts und die von ihnen eingesetzten Schriftfamilien.

Tabelle 11.5 auf S. 365 gibt eine Übersicht über die alten Pakete und die von ihnen installierten Schriften. Der dort aufgeführte Font ‚Utopia‘ wird aus Gründen der Kompatibilität zu früheren Versionen weiter unterstützt, obwohl er wegen seiner Lizenzbedingungen nicht mehr zur Minimal-Distribution von $\text{\LaTeX}$ gehört. Im Benutzerhandbuch Schmidt (2004) von Walter Schmidt wird außerdem darauf hingewiesen, dass

Tabelle 11.1: Aktuelle Pakete für PostScript-Fonts und die von ihnen eingesetzten Schriftfamilien

Paket	Mit Serifen	Serifenlos	Schreibmaschine	Formeln
(none)	CM Roman	CM Sans Serif	CM Typewriter	~ CM Roman
mathpazo	Palatino			~ Palatino
mathptmx	Times			~ Times
helvet		Helvetica		
avant		Avant Garde		
courier			Courier	
chancery	Zapf Chancery			
bookman	Bookman	Avant Garde	Courier	
newcent	New Century	Avant Garde	Courier	
	Schoolbook			
charter	Charter			

das neue Paket fourier erweiterte Möglichkeiten bei der Verwendung dieses Fonts bietet und zusätzlich geeignete mathematische Fonts bereitstellt.

Eine Übersicht über die von psnfss unterstützten PostScript-Schriften gibt Tabelle 11.2. In der Spalte „Familie“ wird der von T_EX beim Laden verwendete Name der Schriftfamilie angegeben. Die zweite Spalte gibt an, für welche Strichstärken Fonts verfügbar sind. m steht für die normale Strichstärke (medium), b für die Fettschrift (bold). Die dritte Spalte gibt die Schnitte an, die zu der Schrift angeboten werden. n steht für die normale aufrechte Schrift, im Folgenden kurz Standardschriftart genannt. sl bezeichnet die geneigte Variante der Schrift. Die Angabe it steht für die kursive Variante der Schrift. sc ist für eine Kapitälchen-Variante der Schrift angegeben. Dabei handelt es sich in der Regel um eine Variante, in der die Kleinbuchstaben durch Verkleinern der Großbuchstaben der Schrift gebildet wurden und damit nicht um „echte“ Kapitälchen, bei denen die Strichstärke für Groß- und Kleinbuchstaben einheitlich sein sollte. Beim Verkleinern der Großbuchstaben verringert sich jedoch die Strichstärke entsprechend. Nur für die Schriften ‚Palatino‘ und ‚Times‘ wurden „echte“ Kapitälchen entwickelt. Die letzte Spalte nennt die vollständigen PostScript-Namen der Schriften, auf denen die in Spalte 3 genannten Schnitte basieren.

Die in Tabelle 11.2 genannten Schriften stehen mit Ausnahme der Symbolfonts jeweils in den T_EX-Kodierungen OT1, T1 und TS1 zur Verfügung. Für die Symbolfonts ‚Symbol‘ und ‚Zapf Dingbats‘ wird die T_EX-Kodierung U verwendet. Das Benutzerhandbuch Schmidt (2004) empfiehlt dringend, bei den Textfonts die Kodierungen T1 und TS1 zu verwenden und in der Präambel die Befehle

Tabelle 11.2: Standardfonts des psnfss-Pakets (nach Schmidt (2004))

Familie	Serie	Schnitt	Name(n) der PostScript-Fonts
<i>Avant Garde</i>			
pag	m	n, sl, sc	AvantGarde-Book, AvantGarde-BookOblique
pag	b	n, sl, sc	AvantGarde-Demi, AvantGarde-DemiOblique
<i>Bookman</i>			
pbk	m	n, sl, it, sc	Bookman-Light, Bookman-LightItalic
pbk	b	n, sl, it, sc	Bookman-Demi, Bookman-DemiItalic
<i>Charter</i>			
bch	m	n, sl, it, sc	CharterBT-Roman, CharterBT-Italic
bch	b	n, sl, it, sc	CharterBT-Bold, CharterBT-BoldItalic
<i>Courier</i>			
pcr	m	n, sl, sc	Courier, CourierOblique
pcr	b	n, sl, sc	Courier-Bold, Courier-BoldOblique
<i>Helvetica</i>			
phv	m	n, sl, sc	Helvetica, Helvetica-Oblique
phv	b	n, sl, sc	Helvetica-Bold, Helvetica-BoldOblique
phv	mc	n, sl, sc	Helvetica-Narrow, Helvetica-Narrow-Oblique
phv	bc	n, sl, sc	Helvetica-Narrow-Bold, Helvetica-Narrow-BoldOblique
<i>New Century Schoolbook</i>			
pnc	m	n, sl, it, sc	NewCenturySchlbk-Roman, NewCenturySchlbk-Italic
pnc	b	n, sl, it, sc	NewCenturySchlbk-Bold, NewCenturySchlbk-BoldItalic
<i>Palatino</i>			
ppl	m	n, sl, it, sc	Palatino-Roman, Palatino-Italic
ppl	b	n, sl, it, sc	Palatino-Bold, Palatino-BoldItalic
<i>Times</i>			
ptm	m	n, sl, it, sc	Times-Roman, Times-Italic
ptm	b	n, sl, it, sc	Times-Bold, Times-BoldItalic
<i>Zapf Chancery</i>			
pzc	m	it	ZapfChancery-MediumItalic
<i>Utopia</i>			
put	m	n, sl, it, sc	Utopia-Regular, Utopia-Italic
put	b	n, sl, it, sc	Utopia-Bold, Utopia-BoldItalic
<i>Symbol</i>			
psy	m	n	Symbol
<i>Zapf Dingbats</i>			
pzd	m	n	ZapfDingbats

```
\usepackage[T1]{fontenc}
\usepackage{textcomp}
```

zu verwenden, da nur so auf alle Zeichen des Fonts zugegriffen werden kann.

Mit Ausnahme des Fonts ‚Utopia‘ enthalten alle Fonts das Euro-Zeichen, das mit dem Befehl `\texteuro` gesetzt werden kann.

Durch die größere Höhe der Buchstaben bei einzelnen Fonts im Vergleich zu den Standardfonts von $\text{\LaTeX}$ kann es daher vorkommen, dass der Zeilenabstand zu eng wird. Da die Festlegung des Zeilenabstands eine Frage des Layouts ist, korrigieren die Pakete dieser Sammlung jedoch nicht. Gegebenenfalls lässt sich der Zeilenabstand mit `\linespread{<Faktor>}` geeignet einstellen.

11.1 Paket avant

Dieses Paket lädt den Font ‚Avant Garde‘ (T $\text{\E}$ X-Fontfamilie pag) als serifenlosen Font (`\sffamily`). Das Paket wird mit

```
\usepackage{avant}
```

geladen. Das Setzen längerer Texte mit diesem Font ist nicht zu empfehlen, da er für plakative Überschriften entwickelt wurde. Der Font ist verfügbar in den Schnitten aufrecht, geneigt und Kapitälchen, jeweils in normaler und fetter Strichstärke. Beim Schnitt Kapitälchen sind die Kleinbuchstaben in der Größe verkleinerte Großbuchstaben und damit keine echten Kapitälchen.

Avant Garde	Auf einem feurigen Rosse floh stolz ein dreister Knabe daher. Da rief ein wilder Stier dem Rosse zu: Schande! von einem Knaben ließ' ich mich nicht regieren! Aber ich, versetzte das Ross. Denn was für eine Ehre könnte es mir bringen, einen Knaben abzuwerfen? (Lessing, <i>Das Ross und der Stier</i>)
-------------	---

11.2 Paket bookman

Das Paket bookman ersetzt alle Schriftfamilien im Textmodus: Als Standardschriftart (`\rmfamily`) wird die Schrift ‚Bookman‘ (TeX-Fontfamilie `pbk`) eingesetzt, als serifenlose Schrift (`\sffamily`) ‚Avant Garde‘ und als Schreibmaschinenschrift (`\ttfamily`) ‚Courier‘. Geladen wird das Paket mit

```
\usepackage{bookman}
```

Die Schriftart ‚Bookman‘ steht in den Schnitten aufrecht, geneigt, kursiv und Kapitälchen zur Verfügung, wobei im Schnitt Kapitälchen die Kleinbuchstaben verkleinerte Großbuchstaben sind (keine echten Kapitälchen). Die serifenlose Schrift ‚Avant Garde‘ ist in 11.1 und die Schreibmaschinenschrift in 11.5 beschrieben.

	Auf einem feurigen Rosse floh stolz ein dreister Knabe daher. Da rief ein wilder Stier dem Rosse zu: Schande! von einem Knaben ließ' ich mich nicht re- gieren!
Bookman	Aber ich, versetzte das Ross. Denn was für eine Ehre könnte es mir bringen, einen Knaben abzuwer- fen?
	(Lessing, <i>Das Ross und der Stier</i>)

11.3 Paket chancery

Das Paket setzt die Schrift ‚Zapf Chancery Italic‘ (TeX-Fontfamilie `pzc`) von Herrmann Zapf als Standardschriftart (`\rmfamily`) ein. Es wird geladen mit dem Befehl

```
\usepackage{chancery}
```

Die Schrift ist nur im kursiven Schnitt in normaler Strichstärke verfügbar.

	<i>Auf einem feurigen Rosse floh stolz ein dreister Knabe daher. Da rief ein wilder Stier dem Rosse zu: Schande! von einem Knaben ließ' ich mich nicht regieren!</i>
Zapf Chancery Italic	<i>Aber ich, versetzte das Ross. Denn was für eine Ehre könnte es mir bringen, einen Knaben abzuwerfen?</i>
	(Lessing, <i>Das Ross und der Stier</i>)

11.4 Paket charter

Der vom Paket als Standardschriftart (`\rmfamily`) eingesetzte Font ‚Bitstream Charter‘ (TeX-Fontfamilie `bch`) gehört nicht zu den 35 Standardfonts. Der Font ‚Charter‘ wurde von der Firma Bitstream zur Verwendung freigegeben. Das Paket wird mit dem Befehl

```
\usepackage{charter}
```

geladen. ‚Charter‘ liegt in den Schnitten aufrecht, geneigt, kursiv und Kapitälchen in normaler und fetter Strichstärke vor. Die Kleinbuchstaben beim Schnitt Kapitälchen sind verkleinerte Großbuchstaben und daher keine echten Kapitälchen.

	Auf einem feurigen Rosse floh stolz ein dreister Knabe daher. Da rief ein wilder Stier dem Rosse zu: Schande! von einem Knaben ließ' ich mich nicht regieren!
Bitstream Charter	Aber ich, versetzte das Ross. Denn was für eine Ehre könnte es mir bringen, einen Knaben abzuwerfen? (Lessing, <i>Das Ross und der Stier</i>)

11.5 Paket courier

Dieses Paket ersetzt die Schreibmaschinenschrift `\ttfamily` durch die Schrift ‚Courier‘ (TeX-Fontfamilie `pcr`). Es wird geladen mit dem Befehl

```
\usepackage{courier}
```

Die ‚Courier‘-Schrift steht in den Schnitten aufrecht, geneigt und Kapitälchen in normaler und fetter Strichstärke zur Verfügung. Der Schnitt Kapitälchen emuliert die Kleinbuchstaben durch verkleinerte Großbuchstaben und ist daher kein echter Kapitälchen-Schnitt.

	Auf einem feurigen Rosse floh stolz ein dreister Knabe daher. Da rief ein wilder Stier dem Rosse zu: Schande! von einem Knaben ließ' ich mich nicht regieren!
Courier	Aber ich, versetzte das Ross. Denn was für eine Ehre könnte es mir bringen, einen Knaben abzuwerfen? (Lessing, <i>Das Ross und der Stier</i>)

11.6 Paket helvet

Dieses Paket installiert den Font ‚Helvetica‘ (T_EX-Fontfamilie phv) als serifenlosen Font. Das Paket wird geladen mit

```
\usepackage[<Skalierung>]{helvet}
```

und erlaubt über das optionale Argument das Anpassen der Größe des Fonts. Die Möglichkeit, den Font zu skalieren, ist wichtig, da unterschiedliche Roman-Schriften, zu denen die ‚Helvetica‘ als serifenloser Font eingesetzt wird, trotz gleicher nominaler Schriftgröße (beispielsweise 10pt) in der Höhe unterschiedlich sind. Ein Beispiel sind die sehr ähnlichen Schriften ‚Times‘ und ‚Palatino‘: die ‚Times‘ in der Schriftgröße 12pt entspricht der ‚Palatino‘ in 11pt. Zu beiden wird in der Regel die ‚Helvetica‘ als serifenlose Schrift eingesetzt. Während die ‚Helvetica‘ in der Größe gut zur ‚Palatino‘ passt, erscheint sie im Vergleich zur ‚Times Roman‘ zu groß.

Als Option kann entweder eine Standardskalierung verlangt oder der genaue Skalierungsfaktor vorgegeben werden:

scaled Lädt die ‚Helvetica‘ in der beispielsweise zur ‚Times Roman‘ passenden Standardskalierung (95 %). Der Skalierungsfaktor ist .95.

scaled=<Skalierungsfaktor>

Lädt die ‚Helvetica‘ mit einem frei wählbaren Skalierungsfaktor. Zur ‚Times Roman‘ passt etwa ein Skalierungsfaktor von .92 sehr gut (scaled=.92)

Die folgende Abbildung 11.1 zeigt die Wirkung der Skalierung am Beispiel der ‚Times Roman‘ kombiniert mit der ‚Helvetica‘.



Abbildung 11.1: ‚Helvetica‘ unterschiedlich skaliert im Vergleich zur ‚Times Roman‘

Helvetica

Auf einem feurigen Rosse floh stolz ein dreister Knabe
daher. Da rief ein wilder Stier dem Rosse zu: Schandel von
einem Knaben ließ' ich mich nicht regieren!

Aber ich, versetzte das Ross. Denn was für eine Ehre
könnte es mir bringen, einen Knaben abzuwerfen?

(Lessing, *Das Ross und der Stier*)

11.7 Paket mathpazo

Mit diesem Paket wird nicht nur der Font ‚Palatino‘ (T_EX-Fontfamilie `ppl`) als Normalfont installiert, sondern es tauscht auch den mathematischen Font gegen einen von Diego Puga aus der ‚Palatino‘ entwickelten virtuellen mathematischen Font aus. In der kursiven Variante der ‚Palatino‘ fehlende mathematische Zeichen wurden aus CM-Fonts ergänzt. Der mathematische Font ist auch in einer fetten Version verfügbar und wird bei mit `\boldmath` gesetzten Formeln oder für fette Buchstaben und Symbole in Formeln verwendet.

Beim Laden des Pakets mit

```
\usepackage[<Option(en)>]{mathpazo}
```

können mehrere Optionen angegeben werden. Durch diese lassen sich spezielle Varianten der ‚Palatino‘ laden. Die folgenden Optionen sind vorgesehen:

sc Durch Angabe dieser Option ist es möglich, die T_EX-Fontfamilie `pplx` als Variante zu laden, die *echte* Kapitälchen enthält. Dies ist eine Besonderheit, weil häufig bei den Kapitälchen die Kleinbuchstaben durch geeignet skalierte Großbuchstaben emuliert werden. Dies gilt in der Regel auch für die PostScript-Fonts.

osf Diese Option lädt eine andere Variante mit dem T_EX-Fontnamen `pplj`, die neben echten Kapitälchen auch Ziffern in der alten Form (oldstyle figures) mit Unterlängen für den Textmodus enthält. Im mathematischen Modus werden die „normalen“ Ziffern verwendet.

slantedGreek

Mit dieser Option werden die griechischen Großbuchstaben im mathematischen Modus in geneigter Schrift (engl. *slanted*) gesetzt. Aufrechte griechische Großbuchstaben stehen ebenfalls zur Verfügung und können mit speziellen Befehlen gesetzt werden.

noBBp1

Auch eine sogenannte ‚Blackboard Bold‘-Variante wird zur Verfügung gestellt, die beispielsweise für die Mengensymbole verwendet werden kann. Sie wird aus den Großbuchstaben der ‚Palatino‘ aufgebaut. Die Option `noBBp1` bewirkt, dass diese Variante nicht aktiviert wird und statt dessen ein anderer Font als ‚Blackboard Bold‘-Variante geladen werden kann.

Um bei Angabe der Option `slantedGreek` aufrechte griechische Großbuchstaben und die ‚Blackboard Bold‘-Variante verwenden zu können, definiert das Paket `mathpazo` auch einige Befehle. Die folgende Liste stellt die Befehle zusammen und beschreibt sie.

`\upGamma` `\upDelta` ... `\upOmega`

Diese Befehle setzen im mathematischen Modus aufrechte griechische Großbuchstaben. Sie können unabhängig davon verwendet werden, ob die Option `slantedGreek` angegeben wurde oder nicht. Aufrechte griechische Kleinbuchstaben stehen nicht zur Verfügung.

`\mathbf{Teilformel}`

Setzt im Gegensatz zum `\mathbf{f}` Variablen in einer Formel in einem fetten kursiven Font. Zahlen in der Teilformel werden fett, aber nicht kursiv gesetzt.

`\mathbb{Großbuchstab(en)}`

Setzt die als Argument angegebenen Großbuchstaben oder die Ziffer 1 in einer ‚Blackboard Bold‘-Schrift, typischerweise Mengensymbole. Außer den Großbuchstaben und der Ziffer 1 enthält dieser Font keine weiteren Zeichen.

`mathpazo` lädt die großen mathematischen Symbole automatisch in der zur Grundgröße der Fonts im Dokument passenden Skalierung. Das normalerweise dazu benötigte `exscale` wird für `mathpazo` daher nicht benötigt.

Vorsicht ist geboten beim mathematischen Font `\mathrm`, der keine griechischen Buchstaben enthält. Versucht man, griechische Buchstaben mit `\mathrm` zu setzen, werden an ihrer Stelle Symbole oder Ligaturen ausgegeben.

Palatino	Auf einem feurigen Rosse floh stolz ein dreister Knabe daher. Da rief ein wilder Stier dem Rosse zu: Schande! von einem Knaben ließ' ich mich nicht regieren! Aber ich, versetzte das Ross. Denn was für eine Ehre könnte es mir bringen, einen Knaben abzuwerfen? (Lessing, <i>Das Ross und der Stier</i>)
----------	---

Den Satz mathematischer Formeln mit diesem Paket zeigt das folgende Beispiel. Das Paket `mathpazo` wird in Zeile 7 mit der Option `slantedGreek` geladen, um in den Zeilen 11 und 12 aufrechte und geneigte griechische Großbuchstaben zeigen zu können. Die Formel in den Zeilen 9 und 10 zeigt Anwendungen des Befehls `\mathbf{d}`. Zeile 13 zeigt eine mathematische Formel in der fetten Version. Die Formel in Zeile 18 zeigt die in ‚Blackboard Bold‘ verfügbaren Zeichen.

```

7 \usepackage[slantedGreek]{mathpazo}
8 \begin{document}
9 \[a = \sqrt{b^2+c^2}, \quad \mathbf{a} = \sqrt{b^2+c^2}, \quad \quad
10 \alpha = \sqrt{\mathbf{\beta}^2 + \mathbf{\gamma}^2} \backslash]
11 \[ \Theta = \Gamma + \Delta, \quad \quad
12 \upTheta = \upGamma + \upDelta \backslash]
13 { \boldmath[x^2=y^2+z^2] }
14 \[i \in \mathbb{N}, \, ; m \in \mathbb{Z}, \, ; x \in \mathbb{R} \backslash]
15 \[ \int_0^x \cos t \, dt = \sin x, \quad \quad
16 \lim_{n \rightarrow \infty} \frac{\log n}{n} = 0 \backslash]
17
18 \[ \mathbb{1} \, ; ABCDEFGHIJKLMNOPQRSTUVWXYZ \backslash]
19 \end{document}

```

$$a = \sqrt{b^2 + c^2}, \quad \mathbf{a} = \sqrt{b^2 + c^2}, \quad \alpha = \sqrt{\beta^2 + \gamma^2}$$

$$\Theta = \Gamma + \Delta, \quad \upTheta = \upGamma + \upDelta$$

$$x^2 = y^2 + z^2$$

$$i \in \mathbb{N}, \, m \in \mathbb{Z}, \, x \in \mathbb{R}$$

$$\int_0^x \cos t \, dt = \sin x, \quad \lim_{n \rightarrow \infty} \frac{\log n}{n} = 0$$

1 ABCDEFGHIJKLMNOPQRSTUVWXYZ

[Buch1/PSnfss/mathpazo.tex]

11.8 Paket mathptmx

Auch das Paket mathptmx ersetzt den Roman-Font und den mathematischen Font. Der Normalfont wird durch den Font ‚Times‘ (T_EX-Fontfamilie ptm) ersetzt, und als Font für den mathematischen Modus wird ein virtueller Font eingesetzt, der von Alan Jeffrey, Sebastian Rathz und Ulrik Vieth auf der Basis der ‚Times‘ entwickelt wurden. Fehlende mathematische Symbole wurden aus ‚Computer Modern‘- und RSFS-Schriften und dem ‚Adobe Symbol‘-Font genommen. Die RSFS-Schriften werden insbesondere für den mathematischen Font `\mathcal` verwendet.

Beim Laden des Pakets mit

```
\usepackage[<option>]{mathptmx}
```

kann durch Angabe der Option `slantedGreek` festgelegt werden, dass die griechischen Großbuchstaben kursiv statt aufrecht gesetzt werden sollen.

Wie beim ‚Palatino‘-Font können auch bei der ‚Times‘ griechische Großbuchstaben unabhängig von der Option stets mit den Befehlen

```
\upGamma \upDelta ... \upOmega
```

gesetzt werden. Aufrechte griechische Kleinbuchstaben stehen auch bei der ‚Times‘ nicht zur Verfügung. Auch `mathptmx` skaliert die großen mathematischen Symbole beim Laden automatisch so, dass sie zur Grundgröße der Fonts im Dokument passen. Auf das Paket `exscale` kann also verzichtet werden.

Ein fatter Font steht im mathematischen Modus nicht zur Verfügung. Von einer Emulation mit Hilfe des Pakets `bm` (s. 9.3, S. 240) rät die Dokumentation ab. Außerdem fehlen die Symbole `\amalg`, `\coprod` und `\jmath`.

	Auf einem feurigen Rosse floh stolz ein dreister Knabe daher. Da rief ein wilder Stier dem Rosse zu: Schande! von einem Knaben ließ' ich mich nicht regieren!
Times	Aber ich, versetzte das Ross. Denn was für eine Ehre könnte es mir bringen, einen Knaben abzuwerfen? (Lessing, <i>Das Ross und der Stier</i>)

Das folgende Beispiel zeigt den Formelsatz mit `mathptmx`, das in Zeile 7 mit der Option `slantedGreek` geladen wird, damit in Zeile 10 aufrechte und geneigte griechische Großbuchstaben gezeigt werden können. Ein fatter mathematischer Font steht nicht zur Verfügung; daher konnten in Zeile 11 nur mit `\mathbf` fette Buchstaben gesetzt werden.

```

7 \usepackage[slantedGreek]{mathptmx}
8 \begin{document}
9 \[a=\sqrt{b^2+c^2}, \quad \alpha=\sqrt{\beta^2+\gamma^2},
10 \quad
11 \quad \Theta=\Gamma+\Delta, \quad \upTheta=\upGamma+\upDelta\]
12 \[i\in \mathbf{N}, \; ; \; m\in \mathbf{Z}, \; ; \; x\in \mathbf{R}\]
13 \[\int_0^x \cos t \, dt = \sin x, \quad
14 \quad \lim_{n\rightarrow\infty} \frac{\log n}{n} = 0\]
15 \end{document}
```

$$a = \sqrt{b^2 + c^2}, \quad \alpha = \sqrt{\beta^2 + \gamma^2}, \quad \Theta = \Gamma + \Delta, \quad \Theta = \Gamma + \Delta$$

$$i \in \mathbf{N}, m \in \mathbf{Z}, x \in \mathbf{R}$$

$$\int_0^x \cos t \, dt = \sin x, \quad \lim_{n \rightarrow \infty} \frac{\log n}{n} = 0$$

[Buch1/PSnfss/mathptmx.tex]

11.9 Paket newcent

Sämtliche Schriftfamilien im Textmodus werden durch das Paket `newcent` ersetzt. Die Schrift ‚New Century Schoolbook‘ (T_EX-Fontfamilie `pnc`) wird als Standardschriftart (`\rmfamily`) eingesetzt. Als serifenlose Schrift (`\sffamily`) wird ‚Avant Garde‘ und als Schreibmaschinenschrift (`\ttfamily`) ‚Courier‘ verwendet. Das Paket wird mit

```
\usepackage{newcent}
```

geladen. Die Schriftart ‚New Century Schoolbook‘ liegt in den Schnitten aufrecht, geneigt, kursiv und Kapitälchen vor. Der Schnitt Kapitälchen verwendet verkleinerte Großbuchstaben als Kleinbuchstaben (keine echten Kapitälchen). Die serifenlose Schrift ‚Avant Garde‘ ist in 11.1 und die Schreibmaschinenschrift in 11.5 beschrieben.

New Century Schoolbook	Auf einem feurigen Rosse floh stolz ein dreister Knabe daher. Da rief ein wilder Stier dem Rosse zu: Schande! von einem Knaben ließ' ich mich nicht regieren! Aber ich, versetzte das Ross. Denn was für eine Ehre könnte es mir bringen, einen Knaben abzuwerfen? (Lessing, <i>Das Ross und der Stier</i>)
------------------------------	---

11.10 Paket pifont

Zu den Standard-PostScript-Fonts gehört auch der Symbolfont ‚Zapf Dingbats‘. Das Paket `pifont` lädt diesen Font und stellt Befehle zur Verfügung, um die Symbole in diesem Font zu verwenden. Zusätzlich stellt das Paket Befehle zur Verfügung, um andere Symbolfonts wie ‚Adobe Symbol‘ nutzen zu können. Da keine Optionen vorgesehen sind, wird das Paket mit

Tabelle 11.3: Symbole im Font ‚Zapf Dingbats‘

32		33		34		35		36		37		38		39	
40		41		42		43		44		45		46		47	
48		49		50		51		52		53		54		55	
56		57		58		59		60		61		62		63	
64		65		66		67		68		69		70		71	
72		73		74		75		76		77		78		79	
80		81		82		83		84		85		86		87	
88		89		90		91		92		93		94		95	
96		97		98		99		100		101		102		103	
104		105		106		107		108		109		110		111	
112		113		114		115		116		117		118		119	
120		121		122		123		124		125		126			
		161		162		163		164		165		166		167	
168		169		170		171		172		173		174		175	
176	⑤	177	⑥	178	⑦	179	⑧	180	⑨	181	⑩	182	①	183	②
184	③	185	④	186	⑤	187	⑥	188	⑦	189	⑧	190	⑨	191	⑩
192	①	193	②	194	③	195	④	196	⑤	197	⑥	198	⑦	199	⑧
200	⑨	201	⑩	202	①	203	②	204	③	205	④	206	⑤	207	⑥
208	⑦	209	⑧	210	⑨	211	⑩	212	→	213	→	214	↔	215	↕
216		217	→	218	↗	219	→	220	→	221	→	222	→	223	→
224		225	→	226	→	227	→	228	→	229	→	230	→	231	→
232	→	233	→	234	→	235	→	236	→	237	→	238	→	239	→
		241	→	242	→	243	→	244	→	245	→	246	→	247	→
248	→	249	→	250	→	251	→	252	→	253	→	254	→		

```
\usepackage{pifont}
```

geladen. Für den Font ‚Zapf Dingbats‘ sind spezielle Befehle vorgesehen. Eine zweite Gruppe von Befehlen dient dem Zugriff auf Zeichen aus anderen Symbolfonts.

11.10.1 ‚Zapf Dingbats‘-Font

Dieser Font wird im Unterschied zu anderen Symbolfonts durch speziell für ihn bestimmte Befehle unterstützt. Die in ihm verfügbaren Symbole zeigt Tabelle 11.3.

Zum Setzen von Zeichen aus dem Font ist der Befehl

```
\ding{Nummer}
```

vorgesehen. `<Nummer>` ist die Nummer des zu setzenden Zeichens laut Tabelle 11.3.

Einige Symbole in diesem Font eignen sich auch gut als Markierungen in `itemize`- und für kleinere `enumerate`-Listen. Eine Anpassung solcher Listen ist allerdings nicht erforderlich, da das Paket für derartige Listen Umgebungen anbietet. Eine Alternative zu `itemize` ist die Listenumgebung `dinglist`

```
\begin{dinglist}{<Nummer>}
  \item...
\end{dinglist}
```

`<Nummer>` ist wie beim Befehl `\ding` die Nummer des als Markierung der Listeneinträge zu verwendenden Symbols. Das folgende Beispiel zeigt eine derartige Liste.

```
7 \usepackage{pifont}
8 \begin{document}
9 \begin{dinglist}{42}
10 \item Erster Listeneintrag
11 \item Zweiter Listeneintrag
12 \item Dritter Listeneintrag
13 \end{dinglist}
14 \end{document}
```

- ☛ Erster Listeneintrag
- ☛ Zweiter Listeneintrag
- ☛ Dritter Listeneintrag

[Buch1/PSnfss/dinglist.tex]

Eine Analogie zu `enumerate`-Listen ist die folgende Umgebung:

```
\begin{dingautolist}{<Nummer>}
  \item...
\end{dingautolist}
```

Hier wird der erste Eintrag durch das Symbol mit der angegebenen Nummer (laut Tabelle 11.3) markiert. Bei jedem weiteren Eintrag wird die Nummer um 1 erhöht und das dieser Nummer zugeordnete Symbol als Markierung verwendet. Jeder Eintrag wird also mit einem unterschiedlichen Symbol markiert. Das kann man als Analogie zu nummerierten Listen sehen, da der Font auch eingekreiste Zahlen von 1 bis 10 in der richtigen Reihenfolge enthält. Damit lassen sich nummerierte Listen mit bis zu 10 Einträgen setzen, die mit eingekreisten Zahlen nummeriert sind. Es stehen sogar 4 verschiedene Varianten derartiger Zahlen zur Verfügung. Das folgende Beispiel zeigt eine Liste, die mit weissen Zahlen in einem schwarzen Kreis nummeriert ist. Verweise auf Listeneinträge werden korrekt umgesetzt und reproduzieren die gesetzten Marken.

```

9  \begin{dingautolist}{202}
10 \item Erstens\label{a}
11 \item Zweitens\label{b}
12 \item Drittens\label{c}
13 \end{dingautolist}
14
15 Die Listeneinträge sind
16 mit \ref{a}, \ref{b} und
17 \ref{c} gekennzeichnet.

```

- ❶ Erstens
- ❷ Zweitens
- ❸ Drittens

Die Listeneinträge sind mit ❶, ❷ und ❸ gekennzeichnet.

[Buch1/PSnfss/dingautolist.tex]

Zwei weitere Befehle können verwendet werden, um Linien zu setzen, die aus einem wiederholten Zeichen aufgebaut werden:

```

\dingfill{<Nummer>}
\dingline{<Nummer>}

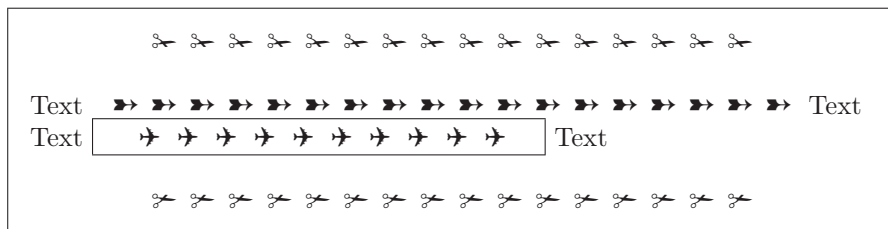
```

Der erste Befehl wirkt ähnlich wie ein `\hline`-Befehl, nur dass der Platz mit dem Symbol des als Arguments angegebenen Nummer gefüllt wird. Der zweite Befehl setzt eine Zeile, in der links und rechts etwas Platz frei bleibt (in der aktuellen Version jeweils 0.5 in) und der Rest der Zeile mit dem als Nummer angegebenen Symbol (nach Tabelle 11.3) gefüllt wird. Das folgende Beispiel zeigt die Wirkung der beiden Befehle:

```

9  \dingline{33}
10
11 \bigskip
12 Text \dingfill{248} Text\
13 Text \framebox[6cm]{\dingfill{40}} Text
14
15 \bigskip
16 \dingline{35}

```



[Buch1/PSnfss/dinglines.tex]

Tabelle 11.4: Symbole im Font ‚Symbol‘

32		33	!	34	∀	35	#	36	∃	37	%	38	&	39	ə
40	(	41	)	42	*	43	+	44	,	45	–	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	≅	65	A	66	B	67	X	68	Δ	69	E	70	Φ	71	Γ
72	H	73	I	74	∅	75	K	76	Λ	77	M	78	N	79	O
80	Π	81	Θ	82	P	83	Σ	84	T	85	Y	86	ς	87	Ω
88	Ξ	89	Ψ	90	Z	91	[	92	∴	93	]	94	⊥	95	–
96		97	α	98	β	99	χ	100	δ	101	ε	102	φ	103	γ
104	η	105	ι	106	φ	107	κ	108	λ	109	μ	110	ν	111	ο
112	π	113	θ	114	ρ	115	σ	116	τ	117	υ	118	ϖ	119	ω
120	ξ	121	ψ	122	ζ	123	{	124		125	}	126	~		
		161	Υ	162	'	163	≤	164	/	165	∞	166	f	167	♣
168	♦	169	♥	170	♠	171	↔	172	←	173	↑	174	→	175	↓
176	◦	177	±	178	”	179	≥	180	×	181	∝	182	∂	183	•
184	÷	185	≠	186	≡	187	≈	188	...	189		190	—	191	↵
192	ℵ	193	ℑ	194	℔	195	℔	196	⊗	197	⊕	198	∅	199	∩
200	∪	201	⊃	202	⊇	203	⊈	204	⊂	205	⊆	206	∈	207	∉
208	∠	209	∇	210	®	211	©	212	™	213	Π	214	√	215	·
216	¬	217	∧	218	∨	219	⇔	220	⇐	221	↑↑	222	⇒	223	↓↓
224	◇	225	⟨	226	®	227	©	228	™	229	Σ	230	(	231	
232	\	233	⌈	234		235	⌋	236	⌈	237	{	238	⌋	239	
		241	⟩	242	⌋	243	⌈	244		245	J	246	)	247	
248	)	249	⌋	250		251	⌋	252	⌋	253	}	254	)		

11.10.2 PostScript-,Symbol'-Font

Die gerade beschriebenen Befehle sind nur Kurzformen für allgemeinere, die die Verwendung beliebiger Symbolfonts erlauben. Bei den allgemeinen Befehlen und Umgebungen ist zusätzlich der Name des Fonts anzugeben, aus dem die Zeichen zu nehmen sind. Für den ‚Dingbat‘-Font ist beispielsweise `pzd` anzugeben oder für den ‚Symbol‘-Font `psy`. Hier wird die Verwendung der Befehle und Umgebungen – bis auf einen Fall – am ‚Symbol‘-Font gezeigt. Die Ausnahme ist ein Smybol aus dem ‚Dingbat‘-Font, auf den auch mit den hier vorgestellten Befehlen und Umgebungen zugegriffen werden kann. Tabelle 11.4 gibt eine Übersicht über die Zeichen und ihre Position im Font, die als Nummer beim Zugriff auf die Zeichen dient.

Dem Kurzbefehl `\ding` entspricht der allgemeine Befehl

```
\Pisymbol{<Font>}{<Nummer>}
```

mit der gleichen Wirkung: er setzt das Symbol mit der als Argument angegebenen Nummer. `<Nummer>` hat die gleiche Bedeutung wie bei `\ding`. Das zusätzliche Argument `<Font>` bezeichnet den Font, aus dem das Zeichen zu nehmen ist. Für einen Font ist dabei sein TeX-Name anzugeben, der beispielsweise der Spalte ‚Familie‘ in Tabelle 11.2 entnommen werden kann. Das Beispiel am Ende dieses Abschnitts zeigt eine Anwendung mit einem ‚Dingbats‘-Symbol.

```
\begin{Pilist}{<Font>}{<Nummer>}
  \item...
\end{Pilist}
```

`<Font>` und `<Nummer>` haben die gleiche Bedeutung wie bei `\Pisymbol`. Es handelt sich um eine Variante der `itemize`-Liste, bei der die Einträge mit dem beim Aufruf angegebenen Symbol gekennzeichnet werden. Das Beispiel zu diesem Abschnitt enthält auch eine `Pilist`-Umgebung.

```
7 \usepackage{pifont}
8 \begin{document}
9 \begin{Pilist}{psy}{196}
10 \item Erster Listeneintrag
11 \item Zweiter Listeneintrag
12 \item Dritter Listeneintrag
13 \end{Pilist}
14 \end{document}
```

- ⊗ Erster Listeneintrag
- ⊗ Zweiter Listeneintrag
- ⊗ Dritter Listeneintrag

[Buch1/PSnfss/pilist.tex]

```
\begin{Piautolist}{<Font>}{<Nummer>}
  \item...
\end{Piautolist}
```

Die Argumente `<Font>` und `<Nummer>` haben die gleiche Bedeutung wie bei der `Pilist`-Liste. Ob man allerdings im Fall des PostScript-, ‚Symbol‘-Fonts noch von einer Analogie zu nummerierten Listen sprechen kann, ist fraglich. Der ‚Symbol‘-Font enthält nämlich als einzige sortierte Symbolfolge die Ziffern von 0 bis 9. Die griechischen Buchstaben sind zwar vorhanden, aber nicht in der Reihenfolge des griechischen Alphabets. Andere Symbolfonts, die ähnlich wie der ‚Zapf Dingbats‘ aufgebaut sind, eignen sich besser für diese Listen. Im Beispiel werden Spielkartensymbole als Marken in einer `Pilist`-Umgebung verwendet.

```

9 \begin{Piautolist}{psy}{167}
10 \item Erster Eintrag
11 \item Zweiter
12     Eintrag\label{a}
13 \item Dritter Eintrag

```

- ♣ Erster Eintrag
- ♦ Zweiter Eintrag
- ♥ Dritter Eintrag

Der zweite Listeneintrag ist mit
♦ gekennzeichnet.

[Buch1/PSnfss/piautolist.tex]

Auch die Befehle `\dingfill` und `\dinglist` sind nur spezielle Formen allgemeinerer Befehle. Die allgemeinen Befehle haben die Form:

```

\Pifill{<Font>}{<Nummer>}
\Piline{<Font>}{<Nummer>}

```

mit den bereits bekannten Argumenten `<Font>` und `<Nummer>`.

Neben diesen Befehlen gibt es einen weiteren, der den Font quasi als Textfont im Textmodus verwendet. Das könnte man beispielsweise beim ‚Symbol‘-Font für griechische Buchstaben nutzen, die bei ihm an den gleichen Positionen stehen wie die Buchstaben in einem Textfont. Der Befehl zum Umschalten in einen Symbolfont lautet

```
\Pifont{<Font>}
```

`<Font>` hat die gleiche Bedeutung wie bisher. Der Befehl entspricht einem Befehl wie `\sfamily` und setzt den Font um, in dem der folgende Text gesetzt wird. Die Wirkung des Befehls kann begrenzt werden, indem er zusammen mit dem in diesem Font zu setzenden Text in geschweifte Klammern eingeschlossen wird. Die griechische Buchstaben im folgenden Beispiel werden mit diesem Befehl erzeugt.

```

9 \Pisymbol{pzd}{168}
10 \Pifill{psy}{172}
11
12 \Pifont{psy}
13 ABCD EFGH IJKL MNOP QRST
14 UVWX YZ
15
16 abcd efgh ijkl mnop qrst
17 uvwx yz
18
19 \Piline{psy}{171}

```

♣ ← ← ← ← ← ← ←
 ΑΒΧΔ ΕΦΓΗ ΙΘΚΛ ΜΝΟΠ ΘΡΣΤ
 ΥΖΩΞ ΨΖ
 αβχδ εφγη ιθκλ μνοπ θρστ υωξ
 ψζ
 ↔ ↔ ↔ ↔

[Buch1/PSnfss/pisonst.tex]

11.11 Veraltete Pakete

Um ältere Dokumente ohne Änderungen weiter formatieren zu können, stehen mittlerweile ersetzte Pakete weiter zur Verfügung, auch wenn von ihrer Verwendung abzuraten ist. Tabelle 11.5 gibt eine Übersicht über diese Pakete und die von ihnen eingesetzten Fonts.

Tabelle 11.5: Veraltete Pakete für PostScript-Fonts und die von ihnen eingesetzten Schriftfamilien

Paket	Mit Serifen	Serifenlos	Schreibmaschine	Formeln
mathpple	Palatino			~ Palatino
mathptm	Times			~ Times
palatino	Palatino	Helvetica	Courier	
times	Times	Helvetica	Courier	
utopia	Utopia			

11.11.1 Paket mathpple

Vorgänger des Pakets mathpazo war dieses Paket. Es installiert ebenfalls die Schrift ‚Palatino‘ als Standardschriftart. Im mathematischen Modus werden wie bei mathpazo virtuelle Fonts eingesetzt, die zu weiten Teilen auf der ‚Palatino‘ basieren. Im Unterschied zu mathpazo stammen die griechischen Buchstaben aus den ‚Euler‘-Fonts, die geneigt wurden, statt aus den ‚Pazo‘-Fonts wie bei mathpazo. Ein ‚Blackboard Bold‘-Font fehlt im mathematischen Modus. Die Varianten der ‚Palatino‘ mit echten Kapitälchen werden nicht unterstützt.

Im mathematischen Modus fehlen die Zeichen `coprod` und fette Varianten von `infty` und `partial`. `jmath` ist dem Font ‚Computer Modern Math Italic‘ entliehen und passt nicht zum ‚Palatino‘-Font.

11.11.2 Paket mathptm

Bei diesem Paket handelt es sich um einen Vorläufer des aktuellen Pakets mathptmx. Als Standardschriftart wird die Schrift ‚Times‘ eingesetzt. Im mathematischen Modus werden hauptsächlich von der ‚Times‘ abgeleitete virtuelle Fonts verwendet. Im Unterschied zum aktuellen Paket mathpazo und dem $\text{\LaTeX}$ -Standard werden griechische Kleinbuchstaben im mathematischen Modus aufrecht statt kursiv gesetzt werden. Als kalligrafischer Font wird im mathematischen Modus ‚Zapf Chancery Italic‘

verwendet. `mathptm` benötigt zusätzlich den Font `,cmex9'`, bei dem allerdings nicht sichergestellt ist, dass er als PostScript Type1 Font installiert ist.

11.11.3 Paket `palatino`

Auch dieses Paket ist ein Vorgänger von `mathpazo` und installiert die Schrift `,Palatino'` als Normalschrift. Zusätzlich werden abweichend vom Paket `mathpazo` als serifenlose Schrift die `,Helvetica'` eingesetzt und als Schreibmaschinenschrift `,Courier'`. Die mathematischen Fonts werden nicht verändert. Falls nicht zusätzlich passende mathematische Fonts geladen werden, werden im mathematischen Modus weiterhin die `,Computer Modern'-Schriften` eingesetzt, und damit andere Schriften als im Text.

11.11.4 Paket `times`

Ein anderer Vorgänger des aktuellen Pakets `mathptmx` ist dieses Paket. Als Standardschriftart wird die `,Times'` eingesetzt. Im Unterschied zum aktuellen Paket werden zusätzlich die `,Helvetica'` als serifenlose Schrift und die `,Courier'` als Schreibmaschinenschrift eingesetzt. Die `,Helvetica'` wird allerdings nicht skaliert geladen und wirkt daher – wie in Abbildung 11.1 auf S. 353 links gezeigt – zu groß. Mathematische Fonts werden nicht geladen. Im mathematischen Modus werden daher weiterhin die `,Computer Modern'-Schriften` verwendet, falls nicht zusätzliche mathematische Fonts angezogen werden. Im mathematischen Modus werden in diesem Fall andere Schriften verwendet als im Textmodus.

11.11.5 Paket `utopia`

Dieses Paket installiert die frei verfügbare PostScript-Schrift `,Utopia'` als Standardschriftart. Andere Schriften – auch die im mathematischen Modus – werden nicht verändert. Die Schrift `,Utopia'` ist auch nicht Teil der Minimalinstallation von $\mathrm{T}_{\mathrm{E}}\mathrm{X}$, da die Lizenzbedingungen nicht vollständig den Richtlinien der `,Free Software Foundation'` genügen. Es kann also nicht davon ausgegangen werden, dass die `,Utopia'` in jeder $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ -Installation verfügbar ist.

Zudem steht das aktuellere und verbesserte Paket `fourier` zur Verfügung, um die `,Utopia'` zu nutzen. Das Paket `fourier` lädt neben anderen Verbesserungen auch passende mathematische Fonts.

Erweiterter Mathematiksatz – Die Pakete der $\mathcal{AMS}$

Mathematischer Formelsatz in hoher typografischer Qualität war von Anfang an ein wichtiger Grund für die Entwicklung von $\text{T}_{\text{E}}\text{X}$ durch Donald E. Knuth. $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ stellt neben einer Reihe von mathematischen Symbolen auch Umgebungen zum Setzen mathematischer Formeln zur Verfügung.

Neben diesen im Standard enthaltenen Symbolen werden in mathematischen Formeln aber eine Vielzahl weiterer Symbole benötigt. Auch bei der Erzeugung von Formeln sind die von $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ angebotenen Umgebungen nicht immer ausreichend, beispielsweise beim Satz mehrzeiliger Formeln oder bei speziellen Matrizendarstellungen.

Aus diesen Gründen hat die $\mathcal{AMS}$ – die American Mathematical Society – ergänzende Pakete entwickelt und zur Verfügung gestellt, die eine große Anzahl weiterer Symbole sowie zusätzliche mathematische Umgebungen und Umgebungen im mathematischen Modus definieren. Für die Symbole wurden spezielle Zeichensätze entwickelt. Dies erleichtert natürlich auch die Eingabe und fördert die Vereinheitlichung von Manuskripten.

Die ersten Entwicklungen der $\mathcal{AMS}$ für $\text{T}_{\text{E}}\text{X}$ begannen 1982. Eine Zeit lang gab es ein eigenständiges Makropaket namens $\mathcal{AMS}\text{T}_{\text{E}}\text{X}$, hauptsächlich entwickelt von Michael Spivak. 1987 wurde das $\mathcal{AMS}\text{-L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Projekt gegründet welches drei Jahre später die Version 1.0 herausbrachte. Damit ist es möglich, auf Basis des Makropakets $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ die Erweiterungen der $\mathcal{AMS}$ zu nutzen. Die Einbindung der Zeichensätze auf Basis des von $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ vorgegebenen Zeichensatzauswahlverfahrens (NFSS) wurde wesentlich von Michael J. Downes vorangetrieben. Die aktuelle Versionsnummer lautet je nach Paket 2.0 bis 2.2, bei kleineren Änderungen noch um einen Buchstaben erweitert.

Die Ergänzungen der $\mathcal{AMS}$ zum Formelsatz bestehen aus den folgenden Paketen:

Pakete: `ambsy`, `amscd`, `amsgen`, `amsmath`, `amsidx`, `amsopn`, `amsrefs`,
`amstex`, `amstext`, `amsthm`, `amsxtra`, `ifoption`, `mathscinet`, `pcatcode`,
`rkeyval`, `textcmds`, `upref`

Pakete mit Befehlen und Umgebungen für den Formelsatz

amsmath

Das zentrale Paket, das die Möglichkeiten zum Satz abgesetzter Gleichungen erweitert und zusätzliche mathematische Konstrukte zur Verfügung stellt. Die Pakete `amstext`, `amsopn` und `ambsy` werden automatisch eingebunden.

amstext

Stellt einen Befehl `\text` zur Verfügung, mit dem in abgesetzte Gleichungen kurze Textpassagen eingefügt werden können.

amsopn

Stellt einen Befehl `\DeclareMathOperator` zur Verfügung, mit dessen Hilfe neue ‚benannte Operatoren‘ entsprechend den vordefinierten `\sin` oder `\lim` definiert werden können.

ambsy

Ein Vorgänger des Pakets `bm` aus den aktuellen Standardpaketen von $\text{\LaTeX}2$, das aus Gründen der Kompatibilität weiterhin zur Verfügung gestellt wird.

amscd

Das Paket `amscd` stellt eine Umgebung `CD` für einfache kommutative Diagramme zur Verfügung, allerdings werden diagonale Pfeile nicht unterstützt.

amsxtra

Stellt einige Erweiterungen wie `\fracwithdelims` und `\accentedsymbol` zur Verfügung, die für eine Kompatibilität mit Dokumenten benötigt werden, die unter der Version 1.1 geschrieben wurden.

amstex

Eine Vorgängerversion des Pakets amsmath, die aus Gründen der Kompatibilität weiterhin zur Verfügung gestellt wird. Da dieses Paket nicht mehr verwendet werden sollte, wird es nicht beschrieben.

Interne Pakete

Interne Pakete sind uninteressant für Anwender und sollten von ihnen nicht geladen werden. Auf diese Pakete wird daher nicht weiter eingegangen.

amsgen

Ein Hilfspaket, das einige von anderen Paketen verwendete interne Befehle zur Verfügung stellt. Es enthält keine dem Anwender zugänglichen Befehle oder Umgebungen. Es wird daher im Folgenden nicht beschrieben und sollte vom Anwender auch nicht geladen werden.

pcatcode

Ein weiteres Hilfspaket von Michael J. Downes, das für Entwickler von Paketen interessant ist, wenn *Catcodes* modifiziert werden, für die ein *Stack* zur Verfügung gestellt wird. Es wird beispielsweise von amsmath verwendet.

rkeyval

Bei diesem Hilfspaket von Michael J. Downes und David M. Jones handelt es sich um eine Reimplementierung des Grundpakets keyval. Im Unterschied zu keyval wird eine restriktivere Syntax verwendet, um eine bessere Fehlerbehandlung zu ermöglichen.

12.1 Paket amsmath

Dieses Paket ist das zentrale Paket für den erweiterten Formelsatz. Pakete zur Definition zusätzlicher mathematischer Symbole und anderer Ergänzungen, die auch getrennt verwendet werden können, werden automatisch angezogen (amsbsy, amsopn und amstext).

Beim Laden des Pakets können Optionen angegeben werden, die einige Einzelheiten des Formelsatzes steuern.

centertags	Die Formelnummer einer mit der <code>split</code> -Umgebung gesetzten mehrzeiligen Formel wird – bezogen auf die gesamte Formel – vertikal zentriert gesetzt (Voreinstellung).
tbtags	Die Formelnummer einer mit der <code>split</code> -Umgebung gesetzten mehrzeiligen Formel wird in die letzte Zeile der Formel gesetzt, wenn die Formelnummer rechts steht, und in die erste Zeile, wenn sie links steht.
sumlimits	Angegebene Exponenten und Indizes werden in abgesetzten Formeln bei Symbolen vom Typ des Summenzeichens $\sum$ – beispielsweise auch bei $\prod$, $\coprod$, $\bigoplus$ und $\bigotimes$ – über und unter das Symbol gesetzt (Voreinstellung). Auf das Integralzeichen hat diese Option keinen Einfluss (vgl. Option <code>intlimits</code>).
nosumlimits	Angegebene Exponenten und Indizes werden stets – auch in abgesetzten Formeln – als Exponenten bzw. Indizes gesetzt.
intlimits	Entspricht der Option <code>sumlimits</code> für das Integralzeichen.
nointlimits	Entspricht der Option <code>nosumlimits</code> für das Integralzeichen (Voreinstellung).
namelimits	Entspricht der Option <code>sumlimits</code> für bestimmte benannte Operatoren wie <code>det</code> , <code>inf</code> , <code>lim</code> , <code>max</code> und <code>min</code> , bei denen diese Schreibweise üblich ist (Voreinstellung).
nonamelimits	Entspricht der Option <code>nosumlimits</code> für die benannten Operatoren.

Die folgenden Optionen werden in der Regel aus der `\documentclass`-Anweisung übernommen, können aber auch beim Laden des Pakets angegeben werden:

leqno	Formelnummern links,
reqno	Formelnummern rechts,
fleqn	Formeln linksbündig mit fester Einrückung.

12.1.1 Übersicht über Formelumgebungen

Zum Setzen von abgesetzten Formeln stehen verschiedene Umgebungen zur Verfügung: Die Reihenfolge der Aufzählung folgt der Komplexität der Formeln, die die Umgebungen enthalten können.

<code>equation</code>	<code>equation*</code>	<code>multline</code>	<code>multline*</code>
<code>gather</code>	<code>gather*</code>	<code>align</code>	<code>align*</code>
<code>alignat</code>	<code>alignat*</code>	<code>flalign</code>	<code>flalign*</code>

Wie üblich verhält sich die `*`-Form der Umgebungen genau so wie die normale Form ohne `*` – aber ohne Vergabe einer Formelnummer. Alle hier aufgeführten Umgebungen beinhalten den Wechsel vom Textmodus in den mathematischen Modus und zurück.

Neben der bereits im Standard- $\text{\LaTeX}$ enthaltenen Umgebung `equation` – hier ergänzt um die Variante `equation*` ohne Formelnummer – sind auch die im Standard enthaltenen Umgebungen `eqnarray` und `eqnarray*` verfügbar. Von der Verwendung der Umgebungen `eqnarray` und `eqnarray*` wird jedoch abgeraten, da eine Kombination der Umgebungen `align` oder `equation` mit der Umgebung `split` entsprechende Möglichkeiten bei besserer Kontrolle über die erzeugte Ausgabe bietet.

Die Standard-Umgebungen `equation`, `eqnarray` und `eqnarray*` sind in den Abschnitten 6.1.2 und 6.1.3 beschrieben. `equation*` ist ein Synonym für die in 6.1.2 beschriebene Standard-Umgebung `displaymath`. Die übrigen Umgebungen obiger Aufzählung werden in Abschnitt 12.1.3 vorgestellt.

Weitere Umgebungen ermöglichen den Satz mehrzeiliger Teilformeln. Diese wechseln nicht selbst in den mathematischen Modus, dazu benötigt man in der Regel eine der oben genannten Umgebungen.

Innerhalb von abgesetzten Formeln kann die Umgebung

```
split
```

verwendet werden, um eine einzelne Formel auf mehrere Zeilen zu verteilen.

Mehrzeilige Teile einer Formel können (ähnlich wie mit der bereits behandelten Umgebung `array`) mit den Umgebungen

```
gathered      aligned      alignedat
```

gesetzt werden. Die Anordnung der Zeilen entsprechen den ähnlich lautenden Umgebungen `gather`, `align` und `alignat`. Um Verwirrung bei den Namen zu vermeiden, ein kleiner Tipp zur Aussprache: Der englische Begriff *align* bedeutet ausrichten. Die Umgebung `alignat`, zu lesen als `align-at`, ist eine Ausrichtungsumgebung ohne horizontalen Abstand. Die Umgebungen für Teilformeln lautet dann analog `align-ed` und `align-ed-at`.

Für Fallunterscheidungen in Formeln steht die spezielle Umgebung

`cases`

zur Verfügung.

Zum Satz von Matrizen im mathematischen Modus bietet `amsmath` die folgenden Alternativen zu `array` an, die teilweise schon vorgegebene Begrenzer wie die runden Klammern bei Matrizen oder die senkrechten Striche bei Determinanten setzen:

<code>matrix</code>	<code>pmatrix</code>	<code>bmatrix</code>	<code>Bmatrix</code>
<code>vmatrix</code>	<code>Vmatrix</code>	<code>smallmatrix</code>	

Alle hier erwähnten Umgebungen für den mehrzeiligen Satz von Teilformeln werden in 12.1.4 genauer behandelt und mit Beispielen vorgestellt.

Die Einbindung des Pakets `amsmath` erfolgt wie üblich mit dem `\usepackage`-Befehl:

```

1 \documentclass[a4paper]{report}
2 \usepackage[T1]{fontenc}
3 \usepackage[ngerman]{babel}
4 \usepackage{amsmath}
5 \setlength{\textwidth}{5cm}
6 \setlength{\parindent}{0cm}
7 \pagestyle{empty}
8 \begin{document}
9 Und hier steht das
10 Dokument, das auch
11 \textsf{amsmath}-Befehle
12 enthalten darf~\dots
13 \end{document}
```

Und hier steht das Dokument, das auch `amsmath`-Befehle enthalten darf ...

[Buch1/AmSPakete/amsmath.tex]

12.1.2 Formelnummerierung

Das Paket amsmath erlaubt eine wesentlich bessere Kontrolle über die Nummerierung der Formeln als der Standard. Es stellt sogar Befehle zur Verfügung, um Symbole oder Text als Kennungszeichen, kurz Kennung, für eine Formel zu verwenden. Neben verbesserten Strategien zur Vermeidung von Überlappungen von Formel und Formelnummer ermöglicht das Paket auch eine bessere Kontrolle über die vertikale Positionierung der Kennung. Unterformeln, engl. *Subequations* werden ebenfalls unterstützt.

12.1.2.1 Kennungen von Formeln und Verweise darauf

Für abgesetzte Formeln mit Nummerierung stellt amsmath die Befehle `\tag` und `\tag*` zur Verfügung, um die Kennzeichnung von Formeln den Erfordernissen anzupassen.

Die Befehle haben die folgende Form:

```
\tag{<Kennung>}
\tag*{<Kennung>}
```

`<Kennung>` steht dabei für die zu vergebende Kennzeichnung der Formel oder Formelzeile und kann neben einfachem Text auch Befehle – darunter der Zugriff auf Zähler – oder mathematische Inline-Formeln enthalten. Dabei ist zu beachten, dass `<Kennung>` im Textmodus und nicht im mathematischen Modus gesetzt wird.

Die Befehle können auch in abgesetzten Formeln ohne Nummerierung verwendet werden, um in Ausnahmefällen doch Formelnummern zu vergeben.

Die Angabe `<Kennung>` wird wie eine echte Formelnummer bei einem Bezug auf die Formel im Text verwendet. Der Unterschied zwischen den beiden Formen ist, dass die `<Kennung>` von `\tag` wie bei Formelnummern üblich in runde Klammern gesetzt wird, während `\tag*` die `<Kennung>` ohne weitere Abgrenzung vom Text setzt. In beiden Fällen liefert ein `\ref`-Befehl den Inhalt von `<Kennung>` als Ergebnis. Wegen eines Programmierfehlers darf der Wechsel in den mathematischen Modus für die `<Kennung>` nur mit `$` und nicht mit `\ ( \)` erfolgen.

```

9 \begin{equation}
10 1+1=2 \tag{e}\label{e}
11 \end{equation}
12 Gleichung~(\ref{e})
13 kennt jedes Kind.
14 \begin{equation}
15 2+2=4 \tag{e$^{\prime}$}
16 \label{e:s}
17 \end{equation}
18 \ref{e:s} doch wohl auch.

```

$$1 + 1 = 2 \quad (e)$$

Gleichung (e) kennt jedes Kind.

$$2 + 2 = 4 \quad (e')$$

e' doch wohl auch.

[Buch1/AmSPakete/tag.tex]

Zum Unterdrücken von Formelnummern in nummerierten Gleichungen sieht das Paket den Befehl

```
\notag
```

vor. Er hat die gleiche Wirkung wie der im Standard verfügbare Befehl `\nonumber`, der ebenfalls verwendet werden kann.

Der `\ref`-Befehl kann wie üblich zum Verweis auf Formelnummern verwendet werden. Er liefert allerdings nur den Inhalt von *<Kennung>*. Runde Klammern, wie sie in der Regel auch bei Verweisen auf Formeln üblich sind, müssen manuell (oder durch ein Makro) hinzugefügt werden. Das Paket `amsmath` stellt für solche Verweise den Befehl `\eqref` zur Verfügung. Er wird ebenfalls im Paket `amstex` definiert.

```
\eqref{<Kennung>}
```

Der Befehl führt zum gleichen Ergebnis wie die Eingabe

```

... (\label{<Marke>})...
... (\ref{<Marke>})...

```

Im Unterschied zum `\tag`-Befehl liefert er den Wert, der mit `\label` markierten aber automatisch nummerierten Formel.

Vor allem für Verweise auf Bereichsangaben von Seiten, beispielsweise 25–39, stellt `amsmath` den Befehl

```
\nobreakdash
```

zur Verfügung, der einen Zeilenumbruch für den (direkt) folgenden Bindestrich verhindert. Der Befehl kann im Textmodus zu diesem Zweck beliebig eingesetzt werden. Er ist zur Verwendung in selbstdefinierten Befehlen geeignet.

Daher zeigt das folgende Beispiel neben einer direkten Verwendung die Definitionen eigener Befehle und ihre Anwendung.

```

9  Vorgegeben sei ein
10 \ (n\)\nobreakdash-%
11 \hspace{0pt}dimensionaler
12 Raum~\dots
13
14 \newcommand{\p}%
15   {\ (p\)\nobreakdash}
16 \p-adisch
17
18 \newcommand{\bis}%
19   {\nobreakdash--\hspace{0pt}}
```

Vorgegeben sei ein n -dimensionaler Raum ...
 p -adisch
 Seiten 25–34,

[Buch1/AmSPakete/nobreakdash.tex]

Das erste Beispiel – offensichtlich ebenfalls besser in Form eines `\newcommand`-Befehls realisiert – zeigt auch, wie beim nachfolgenden Wort die normale Trennung wieder ermöglicht werden kann. Einfacher ist bei Verwendung von `babel` bzw. `ngerman` statt `\hspace{0}` der Befehl `"`.

12.1.2.2 Nummerierung von Formeln in vorgegebenen Abschnitten

Entsprechend den Voreinstellungen werden Formeln in der Dokumentklasse `article` über das gesamte Dokument durchnummeriert. Bei den Klassen `report` und `book` werden sie hingegen kapitelweise nummeriert. Abweichungen vom Standard können in $\text{\LaTeX} 2_{\epsilon}$ vereinbart werden, aber die Vorgehensweise ist – vor allem wegen des Rücksetzens des Formelzählers – verhältnismäßig kompliziert. Eine Automatisierung ist nur unter Verwendung interner $\text{\LaTeX} 2_{\epsilon}$ -Befehle möglich.

Das Paket `amsmath` stellt nun einen einfachen Befehl zur Verfügung, mit dem für beliebige Zähler festgelegt werden kann, welcher Zähler – beispielsweise `subsection` – ihn zurücksetzen soll. Der Befehl hat die Form

```
\numberwithin[<Darstellung>]{<Zähler>}
  {<Rücksetzzähler>}
```

und legt gleichzeitig die Darstellung des Zählers `<Zähler>` so fest, dass der ihn zurücksetzende Zähler `<Rücksetzzähler>` vorangestellt wird. `<Darstellung>` erlaubt es, einen Befehl zur Darstellung des Zählers `<Zähler>` vorzugeben; voreingestellt ist der Befehl `\arabic`, der den Wert als natürliche (arabische) Zahl setzt.

Dieser Befehl ist von generellem Interesse, da er auf jeden Zähler angewendet werden kann. Die Klassen `amsart`, `amsbook` und `amspoc` stellen ihn daher ebenfalls zur Verfügung. Das Paket `amstex` enthält eine vereinfachte Version ohne das optionale Argument `{Darstellung}`.

```

1 \documentclass{amsart}
2 \usepackage[T1]{fontenc}
3 \setlength{\textwidth}{5cm}
4 \begin{document}
5 \thispagestyle{empty}
6 \numberwithin{equation}%
7   {subsection}
8 \section{Section Eins}
9 \subsection{Subsection Eins}
10 \subsection{Subsection Zwei}
11 \begin{equation}
12   c^2=a^2+b^2
13 \end{equation}

```

1. SECTION EINS

1.1. Subsection Eins.

1.2. Subsection Zwei.

$$(1.2.1) \quad c^2 = a^2 + b^2$$

[Buch1/AmSPakete/numberwithin.tex]

12.1.2.3 Vertikale Ausrichtung der Formelnummer

Das Paket `amsmath` tut sein Bestes um zu vermeiden, dass eine Formelnummer eine Formel überschreibt. Falls erforderlich, wird dazu die Formelnummer an einer anderen als der eigentlich für sie vorgesehenen Stelle in den Text gesetzt, und zwar in eine neue Zeile. Die (vertikale) Positionierung der verschobenen Formelnummer ist allerdings kritisch und kann nicht in allen Fällen von einem Programm optisch zufriedenstellend vorgenommen werden. Um dem Setzer eine Einflussmöglichkeit zu bieten, stellt `amsmath` den Befehl

```
\raisetag{<Länge>}
```

zur Verfügung, der ausschließlich verschobene Formelnummern anhebt (im Fall einer positiven `<Länge>`) oder tiefer setzt (bei Angabe einer negativen `<Länge>`). Korrekturen der vertikalen Position von Formelnummern sollten allerdings erst zum Schluss – wenn der Text so gut wie fertig ist – vorgenommen wird, da es sich dabei um Feinabstimmungen des Layouts handelt. In der bei Erstellung des Buchs vorliegenden aktuellen Version von `amsmath` wurde in einer `equation`-Umgebung die Formelnummer bei einer zu langen Formel in eine neue Zeile verschoben, aber der `\raisetag`-Befehl war bedauerlicherweise ohne Wirkung. Nach Auswechseln der `equation`- gegen eine `gather`-Umgebung war die Verschiebung möglich.

Das folgende Beispiel zeigt eine Situation, in der der Einsatz des `\raisetag`-Befehls sinnvoll ist, da durch große Unterlängen im ersten Teil der Formel die wegen Platzmangels in eine neue Zeile gesetzte Formelnummer weit – nach Ansicht der Autoren zu weit – nach unten rutscht. Der `\raisetag`-Befehl wird genutzt, um die Formelnummer deutlich nach oben – praktisch in den ungenutzten Raum der Formelzeile – zu verschieben.

```

9   \begin{gather}
10  F(x,n) = (x-1)(x-3) +
11    \frac{\int_0^x e^{-t^2/2} dt}{\sum_{i=1}^n e^{-x^2/i}}
12    + (h(x) - x)^2 \cdot (g(x) - x^2)
13    + 17x^n \tag{$*$} \\
14    \\
15    \%
16  F(x,n) = (x-1)(x-3) +
17    \frac{\int_0^x e^{-t^2/2} dt}{\sum_{i=1}^n e^{-x^2/i}}
18    + (h(x) - x)^2 \cdot (g(x) - x^2)
19    + 17x^n \tag{$**$}\raisetag{12pt}
20
21  \end{gather}

```

$$F(x,n) = (x-1)(x-3) + \frac{\int_0^x e^{-t^2/2} dt}{\sum_{i=1}^n e^{-x^2/i}} + (h(x) - x)^2 \cdot (g(x) - x^2) + 17x^n \quad (*)$$

$$F(x,n) = (x-1)(x-3) + \frac{\int_0^x e^{-t^2/2} dt}{\sum_{i=1}^n e^{-x^2/i}} + (h(x) - x)^2 \cdot (g(x) - x^2) + 17x^n \quad (**)$$

[Buch1/AmSPakete/raisetag.tex]

12.1.2.4 Unternummerierung von Formeln

Gelegentlich möchte man mehreren Formeln eine gemeinsame Formelnummer geben, aber gleichzeitig in der Lage sein, sich auf die einzelnen Formeln zu beziehen. `amsmath` stellt für solche Fälle die Umgebung `subequations` zur Verfügung. Diese Umgebung muss im Textmodus beginnen und enden. Sie wechselt auch nicht selbst in den mathematischen Modus. Aber alle in dieser Umgebung gesetzten nummerierten Formeln

erhalten eine gemeinsame Formelnummer, an die zur individuellen Kennung ein Buchstabe angehängt wird, tatsächlich aber nur ein Zähler in der Darstellung `\alph`. Nach dem Ende der Umgebung (im Textmodus!) werden Formeln wieder wie vor ihrem Beginn gekennzeichnet. Verwendet wird die Umgebung in der folgenden Form:

```
\begin{subequations}
  [ <Text> ]
  [ \begin{equation} <Formel> \end{equation} ]
  [ \begin{multline} <Formel> \end{multline} ]
  [ \begin{gather} <Formel> \end{gather} ]
  ...
\end{subequations}
```

In dieser Umgebung können Absätze und nummerierte Formeln wie die als Beispiel angegebenen Umgebungen `equation`, `multline` und `gather` in beliebiger Reihenfolge gemischt werden. Der Zähler `equation` ist der individuelle Zähler, der als Buchstabe dargestellt wird. Der allen Kennungen gemeinsame Teil – außerhalb der Umgebung die Formelnummer – ist innerhalb der Umgebung im Zähler `parentequation` abgelegt. Auf beide Zähler können die üblichen Befehle zur Darstellung und Veränderung angewendet werden.

Durch Schachtelung der Umgebung lässt sich die Nummerierungstiefe weiter erhöhen. Durch Ändern der Darstellung des Zählers – des Befehls `\theequation` – ist neben einer anderen Darstellung des Zählers als Buchstabe auch eine Gliederung der Formelnummer mit den üblichen Befehlen möglich.

Das folgende Beispiel zeigt, wie die Umgebung verwendet werden kann und wie die Ausgabe aussieht.

```
9 \begin{subequations}\label{amsmath:Kreis}
10 \subsubsection*{Ein Beispiel f"ur die Anwendung der
11 Umgebung \texttt{subequations}}
12 %
13 Die Gleichung
14 %
15 \begin{equation}\label{amsmath:circle}
16 x^2+y^2=1
17 \end{equation}
18 %
19 beschreibt den Einheitskreis.
20
21 \begin{subequations}
22 \renewcommand{\theequation}%
23 {\theparentequation-\roman{equation}}
```

```

24 \begin{gather}
25 x=\sin t\backslash
26 y=\sin t
27 \end{gather}
28 \end{subequations}
29 %
30 ist die zu (\ref{amsmath:circle}) geh"orende
31 Parameterdarstellung.
32 \end{subequations}
33
34 Die Gleichungen \eqref{amsmath:Kreis} zeigen verschiedene
35 M"oglichkeiten zur Darstellung des Einheitskreises mit
36 Hilfe von reellen Koordinaten. F"ur komplexe Zahlen der
37 Form \((z=x+iy)\) ergibt sich die folgende vereinfachte
38 Formel:
39 %
40 \begin{equation}
41 |z|=1
42 \end{equation}

```

Ein Beispiel für die Anwendung der Umgebung subequations

Die Gleichung

$$x^2 + y^2 = 1 \tag{1a}$$

beschreibt den Einheitskreis.

$$x = \sin t \tag{1b-i}$$

$$y = \sin t \tag{1b-ii}$$

ist die zu (1a) gehörende Parameterdarstellung.

Die Gleichungen (1) zeigen verschiedene Möglichkeiten zur Darstellung des Einheitskreises mit Hilfe von reellen Koordinaten. Für komplexe Zahlen der Form $z = x + iy$ ergibt sich die folgende vereinfachte Formel:

$$|z| = 1 \tag{3}$$

[Buch1/AmSPakete/subequations.tex]

Bei der `subequations`-Umgebung ist vor allem darauf zu achten, dass sie nicht selbst in den mathematischen Modus wechselt, sondern nur die Nummerierung der in ihr gesetzten Formeln verändert. Es handelt sich daher um eine Umgebung zur Veränderung des Layouts und nicht um eine Umgebung für Formeln – auch wenn ihre einzige Wirkung in einer Änderung der Nummerierung von abgesetzten Formeln besteht.

12.1.3 Umgebungen für abgesetzte Formeln

12.1.3.1 Abstände und Zeilenumbrüche

Alle hier behandelten Umgebungen für abgesetzte Formeln sind für mehrzeilige Ausgaben ausgelegt. Wie üblich findet im mathematischen Modus ein automatischer Zeilenumbruch nicht statt. Statt dessen muss der Autor die einzelnen Zeilen in gewohnter Weise mit `\\` abschließen.

Der Zeilenabstand lässt sich durch Angabe eines optionalen Abstands in der Form `\\[<Abstand>]` verändern, wobei ein positiver `<Abstand>` den Zeilenabstand um den angegebenen Wert vergrößert. Entsprechend verringert ein negativer Abstand den vertikalen Zwischenraum der beiden Zeilen. Die letzte Zeile darf nicht mit `\\` abgeschlossen werden, da sonst eine neue Formelzeile begonnen – und gegebenenfalls nummeriert – würde.

Seitenumbrüche in mehrzeiligen Formeln sind in den Standard-Umgebungen nicht möglich. Voreingestellt ist auch beim Paket `amsmath`, dass Seitenumbrüche in mehrzeiligen Formeln verhindert werden. `amsmath` stellt allerdings einen Befehl zur Verfügung, der – am besten direkt vor dem Befehl `\\` angegeben – einen Seitenumbruch nach der aktuellen Formelzeile ermöglicht oder sogar erzwingt. Zusätzlich besteht auch die Möglichkeit, Seitenumbrüche in mehrzeiligen Formelumgebungen durch Angabe des Befehls `\allowdisplaybreaks` in der Präambel des Dokuments grundsätzlich zu erlauben. In diesem Fall kann der im Zusammenhang mit dem Zeilenumbruch behandelte Befehl `\\*` verwendet werden, um einen Seitenumbruch an einer bestimmten Stelle zu verbieten.

Wenn Formelnummern gesetzt werden, versucht das Paket `amsmath` das Überschreiben von Teilen der Formel durch die Formelnummer zu verhindern. Falls erforderlich wird die Formelnummer in eine eigene Zeile gesetzt. Gegebenenfalls kann sie mit dem Befehl `\raisetag` (s. 12.1.2.3) vertikal verschoben werden, um den optischen Eindruck zu verbessern.

12.1.3.2 Befehle im Zusammenhang mit mehrzeiligen Formeln

Der Zeilenumbruch in Umgebungen für mehrzeilige Formeln erfolgt mit dem Befehl `\\[<Länge>]`. Das optionale Argument dieses Befehls kann zur Veränderung des Zeilenabstands verwendet werden. Auch die Sonderform `\\*` steht zur Verfügung, um einen Seitenumbruch in einer Formel an dieser Stelle zu verhindern, falls Zeilenumbrüche in Formeln zugelassen werden.

Bei einer Formel, die über eine größere Anzahl von Zeilen entwickelt wird, kann es erforderlich sein, einen Seitenumbruch auch innerhalb der Formel zuzulassen. Dafür stellt das Paket amsmath einen Befehl `\displaybreak` zur Verfügung, der einen Seitenumbruch nach der Formelzeile, in der er angegeben ist, zulässt. Der Befehl hat die Form

```
\displaybreak[<n>]
```

und hat eine ähnliche Wirkung wie der im Fließtext verwendete Befehl `\pagebreak`.

Bei Verwendung ohne optionales Argument wird ein Seitenumbruch nach der aktuellen Zeile erzwungen. Als optionales Argument kann wie beim `\pagebreak`-Befehl eine ganze Zahl zwischen 0 und 4 angegeben werden. `\displaybreak[0]` lässt einen Seitenumbruch zu, wenn sich nur so eine überlange oder sehr lose – also außerordentlich schlecht – gesetzte Seite vermeiden lässt. `<n> = 4` erzwingt einen Seitenumbruch und ist gleichbedeutend mit dem Fehlen des optionalen Arguments. Werte von 1, 2 oder 3 für `<n>` drücken einen immer stärkeren Wunsch nach einem Seitenumbruch nach der aktuellen Zeile aus, ohne ihn jedoch zu erzwingen.

Will man Seitenumbrüche in Formeln grundsätzlich zulassen, kann in der Präambel der Befehl

```
\allowdisplaybreaks[<n>]
```

verwendet werden. `<n>` ist dabei eine natürliche Zahl zwischen 1 und 4. Bei Angabe des Werts 1 sind Seitenumbrüche in Formeln zwar grundsätzlich erlaubt, werden aber nur im Notfall eingesetzt. Mit steigendem `<n>` ist $\text{\LaTeX}$ zunehmend weniger restriktiv mit Seitenumbrüchen in Formeln. Bei fehlender Angabe von `<n>` wird der höchste Wert angenommen.

Soll in Einzelfällen ein Seitenumbruch nach der aktuellen Zeile ausgeschlossen werden, kann man für den Zeilenumbruch den Befehl `\*` verwenden.

Es wird auch eine Möglichkeit geboten, mehrzeilige Formelumgebungen mit einem kurzen Text zu unterbrechen. Der Befehl dafür hat die Form

```
\intertext{<Text>}
```

und setzt sein Argument als kurzen (fortgesetzten) Absatz an der angegebenen Stelle zwischen die Zeilen. Da es sich nicht um eine Formelzeile handelt, wird der Befehl nicht mit `\\` abgeschlossen. Vielmehr folgt direkt die nächste Formelzeile.

12.1.3.3 Die Umgebungen `multline` und `multline*`

`multline` dient wie die Umgebung `equation` dazu, eine mehrzeilige Formel mit einer einzigen Formelnummer zu setzen. Die Umgebung `multline*` unterscheidet sich von `multline` nur durch die fehlende Nummerierung. Der Unterschied zu den Standard-Umgebungen `equation` und `displaymath` besteht darin, dass sich die Formel über mehrere Zeilen erstrecken kann. Die Zeilen sind wie immer mit Ausnahme der letzten in der gewohnten Weise mit `\\` abzuschließen, da sie nicht automatisch umbrochen werden. Die Umgebungen haben daher die folgende Form:

```
\begin{multline}[*]
  <Erste Formelzeile> [ [ \\
  <Mittlere Formelzeile(n)> ] ... \\
  <Letzte Formelzeile> ]
\end{multline}[*]
```

Die erste Zeile der Formel wird linksbündig gesetzt, die letzte rechtsbündig. Eventuell vorhandene mittlere Zeilen werden zentriert oder bei Angabe der Option `fleqn` mit einer Einrückung linksbündig gesetzt.

Eine rechts von der Formel stehende Formelnummer wird in der letzten Zeile gesetzt (voreingestellt oder Option `reqno`). Im Fall links stehender Formelnummern (Option `leqno`) erscheint sie in der ersten Zeile. Wird eine andere Positionierung der Formelnummer – etwa umgekehrt wie gerade beschrieben oder vertikal zentriert – gewünscht, kann statt der `multline`-Umgebung eine geschachtelte `split/equation`-Umgebung verwendet werden.

```
9 \begin{multline}
10 P(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 + a_4 \cdot x^4 + a_5 \cdot x^5
11         + a_6 \cdot x^6 + a_7 \cdot x^7 + a_8 \cdot x^8 + a_9 \cdot x^9
12         + a_9 \cdot x^9
13
14 \end{multline}
```

$$P(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 + a_4 \cdot x^4 + a_5 \cdot x^5 \\ + a_6 \cdot x^6 + a_7 \cdot x^7 + a_8 \cdot x^8 + a_9 \cdot x^9 \quad (1)$$

[Buch1/AmSPakete/multline1.tex]

```

9  \begin{multline}
10 P(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 \\
11         + a_4 \cdot x^4 + a_5 \cdot x^5 + a_6 \cdot x^6 \\
12         + a_7 \cdot x^7 + a_8 \cdot x^8 + a_9 \cdot x^9
13 \end{multline}

```

$$\begin{aligned}
 P(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 \\
 + a_4 \cdot x^4 + a_5 \cdot x^5 + a_6 \cdot x^6 \\
 + a_7 \cdot x^7 + a_8 \cdot x^8 + a_9 \cdot x^9 \quad (1)
 \end{aligned}$$

[Buch1/AmSPakete/multline2.tex]

```

9  \begin{multline*}
10 P(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 \\
11         + a_4 \cdot x^4 + a_5 \cdot x^5 \\
12         + a_6 \cdot x^6 + a_7 \cdot x^7 + a_8 \cdot x^8 \\
13         + a_9 \cdot x^9
14 \end{multline*}

```

$$\begin{aligned}
 P(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 + a_4 \cdot x^4 + a_5 \cdot x^5 \\
 + a_6 \cdot x^6 + a_7 \cdot x^7 + a_8 \cdot x^8 + a_9 \cdot x^9
 \end{aligned}$$

[Buch1/AmSPakete/multline3.tex]

Speziell für die `multline`-Umgebungen gibt es Befehle, um eine mittlere Formelzeile ganz nach rechts oder ganz nach links zu schieben.

```
\shoveleft{<Formelzeile>}
```

schiebt die gesamte (mittlere) Formelzeile, in der es steht, nach links, während

```
\shoveright{<Formelzeile>}
```

die Zeile nach rechts schiebt. `<Zeile>` ist dabei die komplette Formelzeile ohne das abschließende `\`.

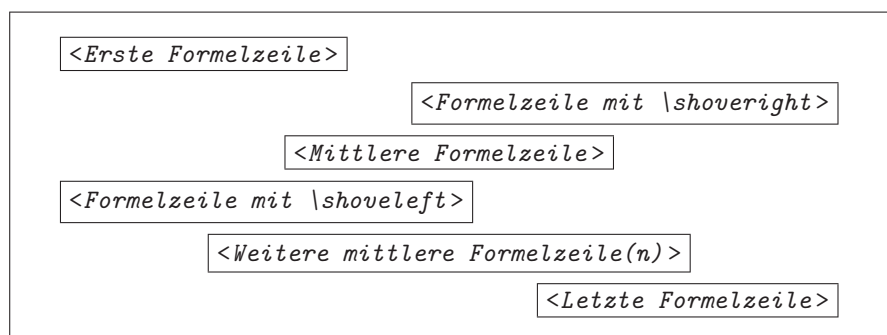
Für die erste und letzte Formelzeile sind diese Befehle nicht vorgesehen. Insbesondere können sie nicht verwendet werden, um diese beiden Zeilen abweichend von der für sie vorgesehenen Ausrichtung zu verschieben.

Damit lassen sich beispielsweise Ausgaben der folgenden Struktur – gezeigt anhand einer schematischen ‚Formel‘ in einer `multline`-Umgebung – realisieren:

```

9  \newcommand{\Bef}[1]{\texttt{\textbackslash#1}}
10 \newcommand{\Parm}[1]{\texttt{<\textit{#1}>}}
11 \begin{multline*}
12  \fbox{\Parm{Erste Formelzeile}}\backslash
13  \shoveright{\fbox{\Parm{Formelzeile mit
14    \Bef{shoveright}}}}\backslash
15  \fbox{\Parm{Mittlere Formelzeile}}\backslash
16  \shoveleft{\fbox{\Parm{Formelzeile mit \Bef{shoveleft}}}}\backslash
17  \fbox{\Parm{Weitere mittlere Formelzeile(n)}}\backslash
18  \fbox{\Parm{Letzte Formelzeile}}
19 \end{multline*}

```



[Buch1/AmSPakete/multline4.tex]

Die Befehle können grundsätzlich in allen mathematischen Umgebungen und auch innerhalb von mehrzeiligen Teilformeln verwendet werden. Speziell in der Umgebung `gathered` für mehrzeilige Teilformeln wirken sie in der gleichen Weise wie bei `multline`. In anderen Umgebungen wirken sich die Verschiebungen nicht vollständig aus oder sind sogar unwirksam.

12.1.3.4 Die Umgebungen `gather` und `gather*`

Diese Umgebungen erlauben es, mehrere Einzelformeln in einer gemeinsamen Umgebung zu setzen. Die Formelzeilen werden unabhängig voneinander horizontal ausgerichtet und – abgesehen von der letzten – mit `\` abgeschlossen. Die Umgebungen entsprechen damit einer direkten Abfolge von `equation`- oder `displaymath`-Umgebungen, bei denen das Beenden und Neubeginnen der Umgebungen zwischen den einzelnen Formeln durch den üblichen Befehl für einen expliziten Zeilenumbruch ersetzt wurde. Die Umgebungen werden wie folgt genutzt:

```
\begin{gather}[*]
  <Erste Formel> [ \\\
  <Weitere Formel(n)> ] ...
\end{gather}[*]
```

In der Umgebung `gather` werden die Formeln nummeriert. Mit einem der Befehle `\notag` oder `\nonumber` lässt sich wie bei der Standard-Umgebung `eqnarray` die Nummerierung einer einzelnen Formelzeile unterdrücken. Bei der Umgebung `gather*` fehlt die Nummerierung vollständig.

```
9  \begin{gather}
10 \sin 2x = 2 \sin x \cos x \\
11 \cos 2x = \cos^2 x
12          - \sin^2 x \\
13 \tan 2x = \frac{2 \tan x}{
14           {1 - \tan^2 x}}
15 \end{gather}
16
17 \begin{gather}
18 \sin 2x = 2 \sin x \cos x \\
19 \cos 2x = \cos^2 x
20          - \sin^2 x \notag \\
21 \tan 2x = \frac{2 \tan x}{
22           {1 - \tan^2 x}}
23 \end{gather}
24
25 \begin{gather*}
26 \sin 2x = 2 \sin x \cos x
27     \tag{$*}$}\label{gat1} \\
28 \cos 2x = \cos^2 x
29          - \sin^2 x \\
30 \tan 2x = \frac{2 \tan x}{
31           {1 - \tan^2 x}}
32     \tag{$**$}\label{gat2}
33 \end{gather*}
34 %
35 Die voranstehenden Formeln
36 der Umgebung \texttt{gather*}
37 haben die Kennungen
38 \eqref{gat1} und \ref{gat2}.
```

$$\sin 2x = 2 \sin x \cos x \quad (1)$$

$$\cos 2x = \cos^2 x - \sin^2 x \quad (2)$$

$$\tan 2x = \frac{2 \tan x}{1 - \tan^2 x} \quad (3)$$

$$\sin 2x = 2 \sin x \cos x \quad (4)$$

$$\cos 2x = \cos^2 x - \sin^2 x$$

$$\tan 2x = \frac{2 \tan x}{1 - \tan^2 x} \quad (5)$$

$$\sin 2x = 2 \sin x \cos x \quad (*)$$

$$\cos 2x = \cos^2 x - \sin^2 x$$

$$\tan 2x = \frac{2 \tan x}{1 - \tan^2 x} \quad **$$

Die voranstehenden Formeln der Umgebung `gather*` haben die Kennungen `(*)` und `**`.

[Buch1/AmSPakete/gather.tex]

12.1.3.5 Die Umgebungen `align` und `align*`

Mit Hilfe dieser Umgebungen ist es möglich, mehrere – im Fall der Umgebung `align` einzeln nummerierte – Formeln an vorgegebenen Stellen horizontal auszurichten, wie das auch bei den Standard-Umgebungen `eqnarray` und `eqnarray*` geschieht. Im Unterschied zu diesen Umgebungen ist die Zahl der Spalten nur durch den verfügbaren Speicher begrenzt. Jeweils zwei Spalten – eine rechtsbündige ohne Abstand gefolgt von einer linksbündigen – bilden eine Gruppe. Zentrierte Spalten wie die mittlere bei der `eqnarray`-Umgebung sind nicht vorgesehen. Die einzelnen Gruppen werden durch einen voreingestellten Abstand getrennt. In den Zeilen noch verfügbarer Platz wird zur Vergrößerung des Abstands zwischen den Gruppen genutzt.

```
\begin{align}[*]
  <r> [ & <l> [ & <r> & <l> ]... [ & <r> ] ] [ \\\
  <r> [ & <l> [ & <r> & <l> ]... [ & <r> ] ] [ ...
\end{align}[*]
```

`<l>` steht dabei für eine linksbündig und `<r>` für eine rechtsbündig ausgerichtete Spalte. Wie bei anderen tabellenartigen Umgebungen können Spalten auch leer sein. Das Zusammenfassen von Spalten mit einem `\multicolumn`-Befehl ist allerdings nicht möglich.

```
9 \begin{align}
10 x &= r\sin\phi & y &= r\cos\phi & & (n=2) \\
11 x &= r\sin\Theta\cos\phi & y &= r\sin\Theta\sin\phi \\
12 & & z &= r\cos\Theta & (n=3) \\
13 \end{align}
```

$x = r \sin \phi$	$y = r \cos \phi$	$(n = 2)$	(1)
$x = r \sin \Theta \cos \phi$	$y = r \sin \Theta \sin \phi$	$z = r \cos \Theta$	$(n = 3) \quad (2)$

[Buch1/AmSPakete/align1.tex]

Wie bei der `gather`-Umgebung lässt sich sich auch bei der Umgebung `align` die Nummerierung von Zeilen mit `\notag` oder `\nonumber` vor dem die Zeile abschließenden `\\` verhindern.

```
9 \begin{align}
10 x &= r\sin\phi & y &= r\cos\phi & & (n=2) \notag \\
11 x &= r\sin\Theta\cos\phi & y &= r\sin\Theta\sin\phi \\
12 & & z &= r\cos\Theta & (n=3) \notag \\
13 \end{align}
```

$$\begin{array}{lll} x = r \sin \phi & y = r \cos \phi & (n = 2) \\ x = r \sin \Theta \cos \phi & y = r \sin \Theta \sin \phi & z = r \cos \Theta \quad (n = 3) \end{array}$$

[Buch1/AmSPakete/align1notag.tex]

12.1.3.6 Die Umgebungen alignat und alignat*

Bei diesen Umgebungen handelt es sich um Varianten der `align`-Umgebungen, bei der die Gruppen ohne Abstand direkt nebeneinander gesetzt werden. Abstände zwischen Gruppen oder Spalten lassen sich nur mit den üblichen Abstandsbefehlen im mathematischen Modus erreichen, die in dieser Umgebung die gleiche Wirkung haben wie auch sonst im mathematischen Modus. Es kommt hinzu, dass die einzelnen Zeilen an den Stellen synchronisiert werden, an denen das Zeichen `&` angegeben ist.

Im Unterschied zu den `align`-Umgebungen muss die Anzahl der Gruppen angegeben werden.

Damit haben die `alignat`-Umgebungen die folgende Form:

```
\begin{alignat}[*]{<ngroups>}
  <r> [ & <l> [ & <r> & <l> ]... [ & <r> ] ] [ \\\
  <r> [ & <l> [ & <r> & <l> ]... [ & <r> ] ] ] ...
\end{alignat}[*]
```

`<l>` und `<r>` stehen wieder für links- bzw. rechtsbündige Spalten, die gegebenenfalls leer bleiben und nicht mit dem `\multicolumn`-Befehl zusammengefasst werden können. `<ngroups>` ist die Anzahl der Gruppen, die sich aus der maximalen Anzahl n des Zeichens `&` in einer Formelzeile nach der Formel

$$\langle ngroups \rangle = n // 2 + 1$$

berechnet, wobei `//` für die ganzzahlige Division steht.

Die folgenden Beispiele zeigen die gleichen Formeln wie bei den `align`-Umgebungen, um den Unterschied deutlich zu machen. Damit die Spalten ähnlich aussehen, wird in jede rechte Spalte ein Abstand mit `\qqquad` eingefügt.

```
9 \begin{alignat}{4}
10 x &= r\sin\phi & \qqquad y &= r\cos\phi \\
11 &\qqquad & \qqquad (n=2) \\
12 x &= r\sin\Theta\cos\phi & y &= r\sin\Theta\sin\phi \\
13 & & z &= r\cos\Theta \quad (n=3) \\
14 \end{alignat}
```

$$\begin{array}{lll} x = r \sin \phi & y = r \cos \phi & (n = 2) \quad (1) \\ x = r \sin \Theta \cos \phi & y = r \sin \Theta \sin \phi & z = r \cos \Theta \quad (n = 3) \quad (2) \end{array}$$

[Buch1/AmSPakete/alignat1.tex]

Wie bei der `gather`-Umgebung lässt sich sich auch bei der Umgebung `alignat` die Nummerierung von Zeilen mit `\notag` oder `\nonumber` vor dem die Zeile abschließenden `\\` verhindern.

Hier folgt das Beispiel mit der `alignat*`-Umgebung, statt `\qqquad` wird jetzt der Abstand `\hspace{2em}` gesetzt, also zweimal die Breite eines ‚M‘ im verwendeten Zeichensatz.

```

9 \begin{alignat*}{4}
10 x &=& r\sin\phi & \& \hspace{2em} y &=& r\cos\phi \\
11 && \& \hspace{2em} & \& \hspace{2em} (n=2) \\
12 x &=& r\sin\Theta\cos\phi & \& y &=& r\sin\Theta\sin\phi \\
13 && \& z &=& r\cos\Theta & \& (n=3) \\
14 \end{alignat*}
```

$$\begin{array}{lll} x = r \sin \phi & y = r \cos \phi & (n = 2) \\ x = r \sin \Theta \cos \phi & y = r \sin \Theta \sin \phi & z = r \cos \Theta \quad (n = 3) \end{array}$$

[Buch1/AmSPakete/alignat2.tex]

12.1.3.7 Die Umgebungen `flalign` und `flalign*`

Bei diesen Umgebungen handelt es sich um eine weitere Variante der `align`-Umgebungen. Der Unterschied besteht im abweichenden Satz der Formelzeilen: Die einzelnen Gruppen werden so weit auseinander gerückt, dass der gesamte verfügbare Platz ausgenutzt wird. Die erste Gruppe stößt also links an den linken und die letzte Gruppe rechts an den rechten Rand des für die Formel verfügbaren horizontalen Platzes an.

```

\begin{flalign}[*]
<r> [ &<l> [ &<r> &<l> ]... [ &<r> ] ] [ &<r> ] ] \\
<r> [ &<l> [ &<r> &<l> ]... [ &<r> ] ] ] ...
\end{flalign}[*]
```

`<l>` steht wiederum für eine linksbündige und `<r>` für eine rechtsbündige Spalte, wobei Spalten durchaus komplett leer sein können. Ein Zusammenfassen von Spalten ist auch hier nicht möglich.

Um die Unterschiede zu den `align`-Umgebungen deutlich zu machen, werden die gleichen Beispiele wie bei diesen Umgebungen gezeigt.

```

9  \begin{flalign}
10 x &= r\sin\phi & y &= r\cos\phi & & (n=2)\\
11 x &= r\sin\Theta\cos\phi & y &= r\sin\Theta\sin\phi \\
12 & & z &= r\cos\Theta & (n=3) \\
13 \end{flalign}

```

$x = r \sin \phi$	$y = r \cos \phi$	$(n = 2) \quad (1)$
$x = r \sin \Theta \cos \phi$	$y = r \sin \Theta \sin \phi$	$z = r \cos \Theta \quad (n = 3) \quad (2)$

[Buch1/AmSPakete/flalign1.tex]

Die Nummerierung von Formelzeilen lässt sich auch hier mit `\notag` oder `\nonumber` vor dem die Zeile abschließenden `\\` verhindern.

Noch einmal das Beispiel mit der `flalign*`-Umgebung.

```

9  \begin{flalign*}
10 x &= r\sin\phi & y &= r\cos\phi & & (n=2)\\
11 x &= r\sin\Theta\cos\phi & y &= r\sin\Theta\sin\phi \\
12 & & z &= r\cos\Theta & (n=3) \\
13 \end{flalign*}

```

$x = r \sin \phi$	$y = r \cos \phi$	$(n = 2)$
$x = r \sin \Theta \cos \phi$	$y = r \sin \Theta \sin \phi$	$z = r \cos \Theta \quad (n = 3)$

[Buch1/AmSPakete/flalign2.tex]

12.1.4 Umgebungen für Teilformeln

Die bisher behandelten Umgebungen setzen mehrzeilige abgesetzte Formeln. Es kommt aber vor, dass in einer einzelnen Formelzeile Teile mehrzeilig zu setzen sind. Dafür steht im Standard die Umgebung `array` zur Verfügung. Das Paket `amsmath` stellt zusätzliche Umgebungen zur Verfügung, analog den gerade behandelten.

<code>split</code>	<code>cases</code>	<code>gathered</code>
<code>aligned</code>	<code>alignedat</code>	

Diese Umgebungen nehmen nicht die gesamte Zeile ein und erlauben daher insbesondere die Verwendung von Klammersymbolen, die die Umgebung einrahmen. Sie können nur im mathematischen Modus verwendet werden. Eine Nummerierung ist nicht vorgesehen, da diese durch die Umgebung erfolgt, in der eine dieser Umgebungen verwendet wird.

Wie bei `array` werden die Teilformeln standardmäßig vertikal zentriert in die Formelzeile gesetzt. Durch Angabe eines optionalen Arguments `b` oder `t` lässt sich aber erreichen, dass die unterste oder oberste Zeile der Teilformel auf der Grundlinie steht.

12.1.4.1 Die Umgebung `split`

Mehrzeilige Teilformeln einer `equation`-Umgebung oder einer Formelzeile können mit der Umgebung `split` gesetzt werden. Zusätzlich bietet `split` die Möglichkeit, die Zeilen an einer mit `&` gekennzeichneten Stelle auszurichten. Im mathematischen Modus wird die `split`-Umgebung in der folgenden Form verwendet:

```
\begin{split}
  <r> [ & <l> ] [ \\\
  <r> [ & <l> ] ] ...
\end{split}
```

Wie bei den `align`-Umgebungen stehen `<l>` und `<r>` für eine links- und eine rechtsbündige Spalte. Gegebenenfalls können Spalten leer bleiben. Auch hier ist das Zusammenfassen von Spalten nicht möglich.

Wegen der Möglichkeit zum Ausrichten der Zeilen an einer mit `&` markierten Stelle wird `split` auch gerne innerhalb einer `equation`-Umgebung statt einer `multline`-Umgebung verwendet. Daher sind

```
9 \begin{equation}
10 \begin{split}
11 \sin x &= \sqrt{1-\cos^2 x} \\
12 &= \frac{\tan x}{\sqrt{1+\tan^2 x}} \\
13 &= \frac{1}{\sqrt{1+\cot^2 x}} \\
14 \end{split}
15 \end{equation}
```

$$\begin{aligned} \sin x &= \sqrt{1 - \cos^2 x} \\ &= \frac{\tan x}{\sqrt{1 + \tan^2 x}} \\ &= \frac{1}{\sqrt{1 + \cot^2 x}} \end{aligned} \quad (1)$$

[Buch1/AmSPakete/split1.tex]

und

```

9 \begin{gather}
10 \sin(x+y) = \sin x \cos y + \cos x \sin y\\
11 \begin{split}
12 \sin(x+y+z) = &\sin x \cos y \cos z + \cos x \sin y \cos z\\
13 &\quad + \cos x \cos y \sin z - \sin x \sin y \sin z
14 \end{split}
15 \end{gather}

```

$$\sin(x+y) = \sin x \cos y + \cos x \sin y \quad (1)$$

$$\begin{aligned}\sin(x+y+z) &= \sin x \cos y \cos z + \cos x \sin y \cos z \\ &\quad + \cos x \cos y \sin z - \sin x \sin y \sin z\end{aligned}\tag{2}$$

[Buch1/AmSPakete/split2.tex]

durchaus typische Beispiele. Auch die folgende Formel kann übersichtlich mit `split` gesetzt werden.

```

9 \begin{equation}
10 \left. \begin{split}
11 \sum_{k=1}^{\infty} \frac{p^k \sin kx}{k!} \\
12 e^{p\cos x} \sin(p\sin x) \backslash \\
13 \sum_{k=0}^{\infty} \frac{p^k \cos kx}{k!} \\
14 e^{p\cos x} \cos(p\sin x) \\
15 \end{split} \right\} \quad \text{quad} \quad \text{right} \backslash \\
16 \quad \quad \quad \text{quad } [p^2 \leq 1] \\
17 \end{equation}

```

$$\left. \begin{aligned} \sum_{k=1}^{\infty} \frac{p^k \sin kx}{k!} &= e^{p \cos x} \sin(p \sin x) \\ \sum_{k=0}^{\infty} \frac{p^k \cos kx}{k!} &= e^{p \cos x} \cos(p \sin x) \end{aligned} \right\} [p^2 \leq 1] \quad (1)$$

[Buch1/AmSPakete/split3.tex]

12.1.4.2 Die Umgebung cases

Das letzte Beispiel zu `split` hat Ähnlichkeiten mit Fallunterscheidungen. Statt der ersten rechtsbündigen Spalte verwendet man bei Fallunterscheidungen aber eher eine linksbündige. Wegen dieser Unterschiede stellt `amsmath` für Fallunterscheidungen im mathematischen Modus die spezielle Umgebung `cases` zur Verfügung, die nicht nur die gewünschten beiden linksbündigen Spalten vorsieht, sondern auch die in diesen Fällen übliche einleitende geschweifte Klammer setzt:

```
\begin{cases}
<1> [ & <1> ] [ \\
<1> [ & <1> ] ] ...
\end{cases}
```

`<1>` steht für die linksbündige Spalte. Bei Bedarf können die beiden Spalten mit einem `\multicolumn`-Befehl zusammengefasst werden.

Das folgende Beispiel zeigt eine Fallunterscheidung, bei der neben den beiden Fällen in einer dritten Zeile der Fallunterscheidung eine Bedingung mit einem `\multicolumn`-Befehl zentriert gesetzt wird.

```
9  \[
10 f(x) =
11 \begin{cases}
12 0 & x \leq 1 \\
13 \log ax & x > 1 \\
14 \multicolumn{2}{c}{(a > 0)}
15 \end{cases}
16 \]
```

$$f(x) = \begin{cases} 0 & x \leq 1 \\ \log ax & x > 1 \\ (a > 0) \end{cases}$$

[Buch1/AmSPakete/cases.tex]

12.1.4.3 Die Umgebung gathered

Mehrzeiliger Satz einer Teilformel ist – im mathematischen Modus – auch in der Form der `gather`-Umgebung möglich. Die Umgebung heißt `gathered` und setzt die Zeilen der Teilformel zentriert:

```
\begin{gathered}
<Erste Zeile der Teilformel> [ \\
<Weitere Teilformel(n)> ] ...
\end{gathered}
```

Die einzelnen Zeilen der Umgebung bestehen aus beliebigen Teilformeln.

Das folgende Beispiel zeigt eine mögliche (einfache) Anwendung der Umgebung `gathered`:

```
9  \[
10 \left.
11 \begin{gathered}
12 \frac{\partial u}{\partial x} \\
13 = \frac{\partial v}{\partial y} \\
14 \frac{\partial u}{\partial y} \\
15 = -\frac{\partial v}{\partial x}
\end{gathered}
\right]
```

```

16 \end{gathered}
17 \quad\right\}
18 \qquad
19 \text{Cauchy-Riemannsche Differentialgleichungen}
20 \}

```

$$\left. \begin{array}{l} \frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x} \end{array} \right\} \quad \text{Cauchy-Riemannsche Differentialgleichungen}$$

[Buch1/AmSPakete/gathered.tex]

Das nachfolgende Beispiel demonstriert das Zusammenspiel einer `split`- und einer `gathered`-Umgebung mit einem kurzen Text dazwischen.

```

9 \begin{equation}
10 \begin{split}
11 \underline{U}_1 = (R_1 + \mathrm{j}X_1) \underline{I}_1 \\
12 \quad + \mathrm{j}X_{\mathrm{h}} \underline{I}_2^{\prime} \\
13 \underline{U}_2^{\prime} = \mathrm{j}X_{\mathrm{h}} \underline{I}_1 \\
14 \quad + (R_2 + \mathrm{j}X_2^{\prime}) \underline{I}_2^{\prime} \\
15 \underline{I}_2^{\prime} \\
16 \end{split}
17 \quad \text{mit} \quad
18 \begin{gathered}
19 X_1 = \omega L_1; X_{\mathrm{h}} = \omega L_{\mathrm{h}} \\
20 X_2^{\prime} = \omega L_2^{\prime} \\
21 \end{gathered}
22 \end{equation}

```

$$\begin{array}{l} \underline{U}_1 = (R_1 + \mathrm{j}X_1)\underline{I}_1 + \mathrm{j}X_{\mathrm{h}}\underline{I}_2' \\ \underline{U}_2' = \mathrm{j}X_{\mathrm{h}}\underline{I}_1 + (R_2 + \mathrm{j}X_2')\underline{I}_2' \end{array} \quad \text{mit} \quad \begin{array}{l} X_1 = \omega L_1; X_{\mathrm{h}} = \omega L_{\mathrm{h}} \\ X_2' = \omega L_2' \end{array} \quad (1)$$

[Buch1/AmSPakete/split.tex]

12.1.4.4 Die Umgebung `aligned`

Mehrzeilige Teilformeln können mit der `aligned`-Umgebung an vorgegebenen Stellen ausgerichtet werden. Wie bei der Umgebung `align` ist die Zahl der Synchronisationsstellen nur durch den verfügbaren Speicher beschränkt.

```
\begin{aligned}
  <r> [ & <l> [ & <r> & <l> ]... [ & <r> ] ] [ \\\
  <r> [ & <l> [ & <r> & <l> ]... [ & <r> ] ] ] ...
\end{aligned}
```

$<l>$ und $<r>$ stehen wieder für links- und rechtsbündige Spalten, die auch leer sein können. Ein Zusammenfassen von Spalten ist nicht möglich.

Als Beispiel werden analog zu `align` von Seite 386 die beiden Formeln (1) und (2) mit nur einer Formelnummer gesetzt.

```
9 \begin{equation}
10 \begin{aligned}
11 x &= r\sin\phi & y &= r\cos\phi & & (n=2) \\
12 x &= r\sin\Theta\cos\phi & y &= r\sin\Theta\sin\phi & z &= r\cos\Theta & (n=3) \\
13 & & & & & \\
14 \end{aligned}
15 \end{equation}
```

$$\begin{array}{llll} x = r \sin \phi & y = r \cos \phi & & (n = 2) \\ x = r \sin \Theta \cos \phi & y = r \sin \Theta \sin \phi & z = r \cos \Theta & (n = 3) \end{array} \quad (1)$$

[Buch1/AmSPakete/aligned.tex]

12.1.4.5 Die Umgebung `alignedat`

Die Ausrichtungsumgebung `alignedat` (zu lesen als align-ed-at) ist eine Entsprechung von `alignat` für Teilformeln. Sie weist gegenüber der Umgebung `aligned` die gleichen Unterschiede auf wie die Umgebung `alignat` gegenüber `align`. Alle Spalten werden ohne zusätzliche Abstände voneinander gesetzt, und die Anzahl der Spaltengruppen – bestehend aus mindestens einer rechtsbündigen Spalte gefolgt von einer linksbündigen – ist als Argument anzugeben:

```
\begin{alignedat}{<ngroups>}
  <r> [ & <l> [ & <r> & <l> ]... [ & <r> ] ] [ \\\
  <r> [ & <l> [ & <r> & <l> ]... [ & <r> ] ] ] ...
\end{alignedat}
```

Auch hier stehen $<l>$ und $<r>$ für links- und rechtsbündige – möglicherweise ungenutzte – Spalten, die nicht zusammengefasst werden können.

Die Anzahl $\langle ngroups \rangle$ der Gruppen berechnet sich wieder aus der Anzahl n der Spalten nach der Formel

$$\langle ngroups \rangle = n // 2 + 1$$

(// bezeichnet die ganzzahlige Division).

Größere (oder kleinere) Abstände zwischen den Spalten lassen sich durch Einfügen von im mathematischen Modus verwendbaren Abständen an geeigneten Stellen erreichen.

Um Gemeinsamkeiten und Unterschiede zu `aligned` deutlich zu machen, wird hier erneut das gleiche Beispiel gesetzt. Die notwendigen Abstände zwischen den Spaltengruppen werden in den linksbündigen Spalten der zweiten Zeile durch den Befehl `\quad` erzeugt.

```

9 \begin{equation}
10 \begin{alignedat}{4}
11 x &= r \sin \phi & y &= r \cos \phi & & (n=2) \\
12 x &= r \sin \Theta \cos \phi & & & & \\
13 & & y &= r \sin \Theta \sin \phi & & \\
14 & & & & z &= r \cos \Theta & (n=3) \\
15 \end{alignedat}
16 \end{equation}
```

$$\begin{array}{llll} x = r \sin \phi & y = r \cos \phi & & (n = 2) \\ x = r \sin \Theta \cos \phi & y = r \sin \Theta \sin \phi & z = r \cos \Theta & (n = 3) \end{array} \quad (1)$$

[Buch1/AmSPakete/alignedat1.tex]

Das nächste Beispiel zeigt eine Formel, bei der `alignedat` verwendet wird, um parallele Teile zweier untereinander stehender Formeln hervorzuheben. Rechts vom Gleichheitszeichen werden dabei nur linksbündige Spalten genutzt.

```

9 \[
10 \begin{alignedat}{3}
11 u(x) &= \sin x & & \\
12 &+ \cos x & & + \tan x \\
13 v(x) &= \arcsin x & & \\
14 &+ \arccos x & & + x \\
15 \end{alignedat}
16 \]
```

$$\begin{array}{llll} u(x) = \sin x & + \cos x & + \tan x \\ v(x) = \arcsin x & + \arccos x & + x \end{array}$$

[Buch1/AmSPakete/alignedat2.tex]

12.1.4.6 Umgebungen für Matrizen, Determinanten, etc.

Auch Matrizen und aus ihrer Darstellung abgeleitete Darstellungen sind mehrzeilige Teilformeln im mathematischen Modus. Parallel zur `array`-Umgebung in Standard- $\text{\LaTeX}$ bietet `amsmath` die Variante `matrix` mit leicht modifizierten Spaltenabständen an. Hinzu kommen Umgebungen, die zusätzlich zur Matrix unterschiedliche Begrenzungen setzen. Die Umgebungen heißen

<code>matrix</code>	<code>pmatrix</code>	<code>bmatrix</code>	<code>Bmatrix</code>
<code>vmatrix</code>	<code>Vmatrix</code>	<code>smallmatrix</code>	

mit den zugehörigen Begrenzern

`()` `[]` `{}` `||` `|||`

Die Umgebung `smallmatrix` steht zum Satz von (kleinen) Matrizen im Fließtext als verkleinerte Variante der `matrix`-Umgebung zur Verfügung.

Im Unterschied zur `array`-Umgebung entfallen bei den Matrix-Umgebungen die Spaltenvereinbarungen. Die maximale Anzahl von Spalten in einer `matrix`-Umgebung wird durch den Zähler `MaxMatrixCols` festgelegt, der mit dem Wert 10 vorbelegt ist. Ohne Veränderung dieses Zählers können Matrizen daher bis zu 10 Spalten enthalten. Bei `matrix` sind die Spalten grundsätzlich zentriert. Eine abweichende Ausrichtung von Spalten ist bei dieser Umgebung nur mit einem `\multicolumn`-Befehl möglich, der auch zum Zusammenfassen von Spalten verwendet werden kann. Da `\multicolumn` allerdings nur lokal wirkt, sollte die `array`-Umgebung verwendet werden, wenn ganze Spalten nicht zentriert werden sollen.

Die Matrix-Umgebungen haben eine einheitliche Form. Ihre Struktur kann daher für sie gemeinsam angegeben werden:

```
\begin{[p|b|B|v|V]matrix}
  <c> [ & <c> ]... [ \
  <c> [ & <c> ]... ]...
\end{[p|b|B|v|V]matrix}
```

Alle Spalten sind zentriert und daher als `<c>` angegeben.

Die gleiche Form hat auch die Umgebung `smallmatrix`, die insbesondere für die Verwendung in Formeln im Fließtext gedacht ist:

```
\begin{smallmatrix}
  <c> [ & <c> ]... [ \\
  <c> [ & <c> ]... ]...
\end{smallmatrix}
```

Im Gegensatz zu den anderen Matrix-Umgebungen entfällt bei `smallmatrix` die Begrenzung der Anzahl der Spalten durch den Zähler `MaxMatrixCols`. Vielmehr ist hier die Spaltenzahl nur durch den verfügbaren Speicher begrenzt. Mit dieser Umgebung können zweizeilige Matrizen wie $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ im Fließtext gesetzt werden, ohne den Zeilenabstand zu ändern. Augenscheinlich ist der normale Zeilenabstand für mehr als zwei Zeilen nicht ausreichend, so dass in diesem Fall ein größerer Zeilenabstand erforderlich ist.

Zunächst sollen die unterschiedlichen Spaltenabstände der Umgebung `matrix` gegenüber `array` an einem Beispiel veranschaulicht werden. Dazu eignet sich am Besten eine Determinante, bei der die Umgebung `matrix` mit Begrenzern verwendet wird, so wie es bei `array` erforderlich ist.

```
9 \begin{align*}
10 \left| \begin{array}{ccc} a&b&c\\
11 -b&a&d\\ -c&-d&a \end{array} \right|
12 &\quad \texttt{array}
13 \left| \begin{matrix} a&b&c\\
14 -b&a&d\\ -c&-d&a \end{matrix} \right|
15 &\quad \texttt{matrix}
```

$\left \begin{array}{ccc} a & b & c \\ -b & a & d \\ -c & -d & a \end{array} \right $	$\left \begin{matrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{matrix} \right $	$\quad \texttt{array} \qquad \qquad \texttt{matrix}$
--	---	--

[Buch1/AmSPakete/matrix1.tex]

Nun der Vergleich von `matrix` mit der Umgebungs-Variante `vmatrix`:

```
9 \begin{align*}
10 \left| \begin{matrix} a&b&c\\
11 -b&a&d\\ -c&-d&a \end{matrix} \right|
12 &\quad \texttt{matrix}
13 \left| \begin{vmatrix} a&b&c\\
14 -b&a&d\\ -c&-d&a \end{vmatrix} \right|
15 &\quad \texttt{vmatrix}
```

$$\begin{vmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{vmatrix} \quad \text{matrix} \qquad \begin{vmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{vmatrix} \quad \text{vmatrix}$$

[Buch1/AmSPakete/matrix2.tex]

Es ist offensichtlich, dass die Umgebung `vmatrix` lediglich eine Abkürzung für `\left|\begin{matrix}...\end{matrix}\right|` ist.

Das folgende Beispiel stellt die verschiedenen Varianten der Matrix-Umgebungen vor:

```

9  \begin{align*}
10 \begin{matrix} \begin{matrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{matrix} \end{matrix} \\
11 \quad \quad \quad \text{matrix} \\
12 \begin{matrix} \begin{pmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{pmatrix} \end{matrix} \\
13 \quad \quad \quad \text{pmatrix} \\
14 \begin{matrix} \begin{bmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{bmatrix} \end{matrix} \\
15 \quad \quad \quad \text{bmatrix} \\
16 \begin{matrix} \begin{Bmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{Bmatrix} \end{matrix} \\
17 \quad \quad \quad \text{Bmatrix} \\
18 \begin{matrix} \begin{vmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{vmatrix} \end{matrix} \\
19 \quad \quad \quad \text{vmatrix} \\
20 \begin{matrix} \begin{Vmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{Vmatrix} \end{matrix} \\
21 \quad \quad \quad \text{Vmatrix} \\
22 \multicolumn{4}{c}{\begin{smallmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{smallmatrix}} \\
23 \quad \quad \quad \text{smallmatrix} \\
24 \end{align*}
25 \end{align*}

```

$$\begin{matrix} \begin{matrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{matrix} \end{matrix} \quad \text{matrix} \qquad \begin{pmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{pmatrix} \quad \text{pmatrix}$$

$$\begin{bmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{bmatrix} \quad \text{bmatrix} \qquad \begin{Bmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{Bmatrix} \quad \text{Bmatrix}$$

$$\begin{vmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{vmatrix} \quad \text{vmatrix} \qquad \begin{Vmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{Vmatrix} \quad \text{Vmatrix}$$

$$\begin{smallmatrix} a & b & c \\ -b & a & d \\ -c & -d & a \end{smallmatrix} \quad \text{smallmatrix}$$

Sehr hilfreich ist auch der von amsmath zur Verfügung gestellte Befehl `\hdotsfor`, der eine vorgegebene Anzahl von Spalten ab der aktuellen Position mit Punkten füllt. Er entspricht in gewisser Weise einem `\multicolumn`-Befehl, wobei entsprechend seiner Aufgabe die Argumente zur Festlegung der Spaltenausrichtung und für den zu setzenden Inhalt fehlen. Damit hat der Befehl die folgende Form:

```
\hdotsfor[<Faktor>]{<ncols>}
```

`<ncols>` ist dabei die Anzahl der Spalten ab der aktuellen Position, die mit Punkten gefüllt werden sollen. Das optionale Argument `<Faktor>` ist ein Vorfaktor des Abstands der Punkte, für den der Wert 1 voreingestellt ist. Kleinere Werte als 1 verringern also den Punktabstand, größere erhöhen ihn.

Der Befehl `\hdotsfor` kann auch in einer `array`-Umgebung genutzt werden. Die Verwendung in einer `tabular`-Umgebung scheitert jedoch daran, dass der Befehl nur im mathematischen Modus korrekt arbeitet. In der Umgebung `smallmatrix` kann der `\hdotsfor`-Befehl ebenfalls nicht eingesetzt werden.

Ein Beispiel für die Verwendung des `\hdotsfor`-Befehls ist die schematische Darstellung einer Matrix:

```

9  \begin{gather*}
10 \begin{pmatrix}
11 a_{11} & a_{12} & \hdotsfor{1} & a_{1j} \\
12 & & & \hdotsfor{1} & a_{1n} \\
13 a_{21} & a_{22} & \hdotsfor{1} & a_{2j} \\
14 & & & \hdotsfor{1} & a_{2n} \\
15 & & \rule{7mm}{0mm} & & \\
16 & & & \rule{7mm}{0mm} \\
17 \hdotsfor{6} \\
18 a_{i1} & a_{i2} & \hdotsfor{1} & a_{ij} \\
19 & & & \hdotsfor{1} & a_{in} \\
20 \hdotsfor{6} \\
21 a_{m1} & a_{m2} & \hdotsfor{1} & a_{mj} \\
22 & & & \hdotsfor{1} & a_{mn} \\
23 \end{pmatrix}
24 \end{gather*}
25
26 \begin{gather*}
27 \begin{pmatrix}
28 a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1n} \\
29 a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2n} \\
30 \hdotsfor{6}

```

```

31 a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{in} \\
32 \hdotsfor{6} \\
33 a_{m1} & a_{m2} & \dots & a_{mj} & \dots & a_{mn} \\
34 \end{pmatrix} \\
35 \end{gather*}

```

$$\begin{pmatrix}
 a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1n} \\
 a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2n} \\
 \hdotsfor{6} \\
 a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{in} \\
 \hdotsfor{6} \\
 a_{m1} & a_{m2} & \dots & a_{mj} & \dots & a_{mn}
 \end{pmatrix}$$

$$\begin{pmatrix}
 a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1n} \\
 a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2n} \\
 \hdotsfor{6} \\
 a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{in} \\
 \hdotsfor{6} \\
 a_{m1} & a_{m2} & \dots & a_{mj} & \dots & a_{mn}
 \end{pmatrix}$$

[Buch1/AmSPakete/hdotesfor.tex]

An der Eingabe der ersten Matrix in diesem Beispiel sind besondere Vorkehrungen zu erkennen, um die dritte und fünfte Spalte mit den Fortsetzungspunkten zu erhalten. Diese beiden Spalten enthalten nur den `\hdotsfor`-Befehl. Da eine Spalte nur so breit gesetzt wird, wie ihr Inhalt vorgibt, der `\hdotsfor`-Befehl jedoch nur so viele Punkte setzt, wie die Spalte breit ist, würden beim Streichen der dritten Matrixzeile, also die Eingabezeilen 15 und 16,

```
\rule{7mm}{0mm}
```

die dritte und fünfte Spalte entfallen. Das liegt daran, dass `\hdotsfor` keine eigene Breite hat. Der zusätzliche negative Zeilenvorschub korrigiert den normalen, hier unerwünschten Zeilenabstand. Würde man die `\rule`-Befehle beispielsweise in die erste Matrixzeile mit einbauen, könnten die Zeilen 15 und 16 entfallen, allerdings leidet die Übersichtlichkeit darunter.

Bei der unteren Matrix wird der `\hdotsfor`-Befehl nur an den Stellen verwendet, bei denen mehr als eine Spalte mit Punkten zu füllen ist. Im Fall einer einzelnen Spalte wird statt dessen der `\dots`-Befehl verwendet. Bei genauem Hinschauen ist auch erkennbar, dass in den Spalten drei und fünf der unteren Matrix der Standardabstand der Punkte bei

`\hdotsfor` dem Abstand der Punkte im `\dots`-Befehl entspricht, während bei der oberen Matrix Anzahl und Abstand der Punkte von der durch den `\rule`-Befehl vorgegebenen Breite abhängen.

12.1.5 Zusätzliche Befehle im mathematischen Modus

Neben den Umgebungen definiert das Paket amsmath eine Reihe von nützlichen Befehlen im mathematischen Modus.

12.1.5.1 Wurzeln

Im Zusammenhang mit dem Wurzelzeichen stellt amsmath zwei Befehle zur Verfügung, die es erlauben, das optionale Argument, das über den Ansatz des Wurzelzeichens geschrieben wird (etwa k für die k -Wurzel) zu verschieben. Der eine Befehl bewirkt eine horizontale Verschiebung, der andere eine vertikale. Die Befehle haben die Form

```
\leftroot{<Zahl>}
\uproot{<Zahl>}
```

mit positiven oder negativen Zahlen $\langle Zahl \rangle$. Positive Werte verschieben nach links bzw. oben, negative nach rechts oder unten. Der angegebene Zahlwert ist das gewünschte Vielfache der mathematischen Längeneinheit μ ($\frac{1}{18}$ em) und erlaubt sehr feines Positionieren des optionalen Arguments. Die Befehle sind als Teil des optionalen Arguments – vor der zu setzenden Teilformel – anzugeben.

Den Unterschied in der Positionierung ohne und mit Angabe einer Verschiebung zeigt das folgende Beispiel.

```
9 \[
10 \sqrt{\beta}
11 {\frac{x+y}{x^2+y^2}}
12 \quad
13 \sqrt{\leftroot{2}\beta}
14 \quad \sqrt{\uproot{4}\beta}
15 {\frac{x+y}{x^2+y^2}}
```

[Buch1/AmSPakete/roots.tex]

12.1.5.2 Bruchähnliche Darstellungen

Das Paket `amsmath` erweitert die Möglichkeiten beim Setzen von Brüchen. Zum einen gibt es zwei zusätzliche Varianten, die unabhängig vom aktuellen mathematischen Modus Brüche wie im Textmodus oder wie in abgesetzten Formeln setzen. Zum anderen ist für Kettenbrüche ein eigener Befehl vorhanden.

<code>frac</code>	<code>dfrac</code>	<code>tfrac</code>	<code>cfrac</code>
<code>binom</code>	<code>dbinom</code>	<code>tbinom</code>	
<code>genfrac</code>			

Das frühere $\mathcal{AMSTeX}$ erweiterte den `\frac`-Befehl um ein optionales Argument zum Einstellen der Stärke des Bruchstrichs. Diese Erweiterung ist entfallen zugunsten eines sehr allgemeinen Befehls, der nicht nur die Vorgabe der Stärke des Bruchstrichs erlaubt, sondern zugleich Begrenzer (Klammern) um den Bruch setzt und wahlweise den Satz im Stil einer Inline-Formel oder einer abgesetzten Formel erlaubt. Diesen Befehl nutzt `amsmath` etwa, um einen Befehl für Binomialkoeffizienten zur Verfügung zu stellen, die wie ein Bruch mit fehlendem Bruchstrich (Strichstärke 0mm) in runden Klammern gesetzt werden. Natürlich werden auch die Brüche selbst über diesen Befehl definiert.

12.1.5.3 Brüche

Standard- $\mathcal{L}^{\text{A}}\mathcal{T}_{\text{E}}\mathcal{X}$ stellt für Brüche (engl. fraction) den bereits beschriebenen Befehl `\frac` zur Verfügung (vgl. 6.2.4). Auch mit `amsmath` ist dies der Befehl für einen Bruch. Zusätzlich bietet `amsmath` die beiden Befehle

```
\dfrac{<Zähler>}{<Nenner>}
\tfrac{<Zähler>}{<Nenner>}
```

an, die im Aufruf vollständig mit dem `\frac`-Befehl übereinstimmen. Der Unterschied liegt darin, dass `\tfrac` den Bruch in der Form setzt, die im Textmodus üblich ist, also mit verkleinertem Zähler und Nenner (text-style), während `\dfrac` die in abgesetzten Formeln gebräuchliche Darstellung benutzt (displaystyle).

Auf den ersten Blick scheinen diese Varianten von recht geringer Bedeutung zu sein. Es ist aber ein großer Vorteil, in Doppel- und Mehrfachbrüchen die Möglichkeit zu haben, zwischen der normalen und verkleinerten Darstellung wählen zu können.

Das folgende Beispiel zeigt die Befehle `\dfrac` und `\tfrac` im Vergleich zu `\frac`.

```

10 Ein Bruch \(\frac{x-y}{x+y}\) im Textmodus \dots
11
12 und in abgesetzten Formeln:
13 \[
14 \frac{x-y}{x+y}\quad (\texttt{\TB frac})
15 \quad\quad\quad
16 \dfrac{x-y}{x+y}\quad (\texttt{\TB dfrac})
17 \quad\quad\quad
18 \tfrac{x-y}{x+y}\quad (\texttt{\TB tfrac})
19 \]
```

Ein Bruch $\frac{x-y}{x+y}$ im Textmodus ...
und in abgesetzten Formeln:

$$\frac{x-y}{x+y} \quad (\texttt{\frac}) \quad \frac{x-y}{x+y} \quad (\texttt{\dfrac}) \quad \frac{x-y}{x+y} \quad (\texttt{\tfrac})$$

[Buch1/AmSPakete/frac1.tex]

Das Beispiel zeigt auch, dass in den hier vorgestellten Fällen die in amsmath definierten Befehle nicht erforderlich sind, da der $\text{\LaTeX}$ -Standard ohnehin die gewünschte Form liefert. Erst bei Mehrfachbrüchen zeigen sich die Vorteile:

```

9 \[
10 \frac{\frac{x+y}{x^2+y^2}}{1 + \frac{a-b}{x + \frac{a+b}{(a-b)^2}}}
11 \quad\quad\quad
12 \frac{\dfrac{x+y}{x^2+y^2}}{1 + \dfrac{a-b}{x + \tfrac{a+b}{(a-b)^2}}}
13 \quad\quad\quad
14 \frac{\tfrac{x+y}{x^2+y^2}}{1 + \tfrac{a-b}{x + \tfrac{a+b}{(a-b)^2}}}
15 \quad\quad\quad
16 \frac{\tfrac{x+y}{x^2+y^2}}{1 + \tfrac{a-b}{x + \tfrac{a+b}{(a-b)^2}}}
17 \]
```

$$\frac{\frac{x+y}{x^2+y^2}}{1 + \frac{a-b}{x + \frac{a+b}{(a-b)^2}}} \quad \frac{\frac{x+y}{x^2+y^2}}{1 + \frac{a-b}{x + \frac{a+b}{(a-b)^2}}}$$

[Buch1/AmSPakete/frac2.tex]

Wie man sieht, ist das zweite Beispiel erheblich leichter lesbar. Der Vollständigkeit halber sei angemerkt, dass Standard- $\text{\LaTeX}$ die Befehle `\displaystyle` und `\textstyle` anbietet; die hier eingeführten Befehle von amsmath erleichtern jedoch das Setzen komplexer Brüche.

12.1.5.4 Kettenbrüche

Für Kettenbrüche (engl. continued fraction) gibt es einen eigenen Befehl, um die spezielle Formatierung solcher Brüche zu unterstützen. Der Befehl hat die gleiche Form wie die übrigen Befehle zum Setzen von Brüchen:

```
\cfrac{<Zähler>}{<Nenner>}
```

Da die Form mit den anderen Brüchen übereinstimmt, wird hier nur noch ein Beispiel für die Verwendung des Befehls gezeigt:

```
9  \[
10 \arctan x = \cfrac{x}{1 + \cfrac{x^2}{3 + \cfrac{4x^2}{5 + \cfrac{9x^2}{7 + \dots}}}}
11 +\cfrac{x^2}{3}
12 +\cfrac{4x^2}{5}
13 +\cfrac{9x^2}{7}
14 +\dotsb}}
15 \]
```

$$\arctan x = \frac{x}{1 + \frac{x^2}{3 + \frac{4x^2}{5 + \frac{9x^2}{7 + \dots}}}}$$

[Buch1/AmSPakete/cfrac.tex]

12.1.5.5 Binomialkoeffizienten

Eine ähnliche Grundform wie Brüche haben auch Binomialkoeffizienten. Zwar fehlt ein Bruchstrich, aber die Positionierung der beiden Ausdrücke entspricht der von Zähler und Nenner eines Bruchs. Außerdem sind Binomialkoeffizienten in runde Klammern eingeschlossen.

amsmath stellt nun für Binomialkoeffizienten entsprechende Befehle wie für Brüche zur Verfügung:

```
\binom{<Oben>}{<Unten>}
\dbinom{<Oben>}{<Unten>}
\tbinom{<Oben>}{<Unten>}
```

`\binom` ist wie `\frac` eine Darstellung, die vom aktuellen mathematischen Modus abhängt. `\tbinom` liefert die im Textmodus verwendete Darstellung von `\binom`, `\dbinom` die Darstellung in abgesetzten Formeln. Die beiden Varianten `\tbinom` und `\dbinom` sind wie die entsprechenden Varianten bei Brüchen vor allem bei Schachtelungen von Bedeutung.

```

9   Ein Binomialkoeffizient
10  \(\binom{n+1}{k}\)
11  im Textmodus \dots
12
13  \newcommand{\TB}%
14    {\textbackslash}
15  und in abgesetzten Formeln
16  \begin{align*}
17    \binom{n+1}{k} \qquad
18    & \texttt{\TB binom} \\
19    \dbinom{n+1}{k} \qquad
20    & \texttt{\TB dbinom} \\
21    \tbinom{n+1}{k} \qquad
22    & \texttt{\TB tbinom}
23  \end{align*}

```

Ein Binomialkoeffizient $\binom{n+1}{k}$ im
Textmodus ...

und in abgesetzten Formeln

$$\begin{array}{ll} \binom{n+1}{k} & \texttt{\TB binom} \\ \dbinom{n+1}{k} & \texttt{\TB dbinom} \\ \tbinom{n+1}{k} & \texttt{\TB tbinom} \end{array}$$

[Buch1/AmSPakete/binom.tex]

Wegen der Ähnlichkeit mit den Brüchen sollen auch hier kurze Beispiele zu den verschiedenen Varianten den Gebrauch demonstrieren.

```

9   \begin{gather}
10  \binom{\binom{n}{m}}{\binom{n+k}{m+k}}
11  \end{gather}
12
13  \begin{gather}
14  \frac{\binom{n+k}{m+k}}{1+\binom{n}{k+\binom{n+k}{m}}}
15  \qquad
16  \frac{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\tbinom{n+k}{m}}}
17  \qquad
18  \frac{\binom{n+k}{m+k}}{1+\dbinom{n}{k+\tbinom{n+k}{m}}}
19  \end{gather}

```

$$\left(\frac{\binom{n}{m}}{\binom{n+k}{m+k}} \right) \quad (1)$$

$$\frac{\frac{\binom{n+k}{m+k}}{1+\binom{n}{k+\binom{n+k}{m}}}}{1+\left(\frac{\binom{n+k}{m+k}}{k+\binom{n}{m}}\right)} \quad \left(1+\frac{\binom{n+k}{m+k}}{k+\binom{n}{m}}\right) \quad (2)$$

[Buch1/AmSPakete/binomex.tex]

Das erste Beispiel lässt erkennen, dass die Darstellung zweifach geschachtelter Binomialkoeffizienten mit `\binom` gut dargestellt wird. `\dbinom` und `\tbinom` finden daher eher in Schachtelungen mit Brüchen An-

wendung. Das linke Beispiel in der zweiten Zeile zeigt, warum der Gebrauch von `\tbinom` und `\dbinom` innerhalb eines Bruchs statt in einer Schachtelung von `\binom` sinnvoll ist: Die runden Klammern werden nicht in der gewünschten Weise vergrößert. Da derartige Verschachtelungen bei Binomialkoeffizienten im Gegensatz zu Brüchen kaum vorkommen, wie es das rechte Beispiel dieser Zeile vorgaukelt, ist dies kein echter Nachteil.

12.1.5.6 Selbstdefinierte Bruch-artige Darstellungen

Die gerade behandelten Befehle `\frac`, `\dfrac`, `\tfrac`, `\binom`, `\dbinom` und `\tbinom` werden von `amsmath` alle über einen einzigen generischen Befehl für Bruch-artige Darstellungen definiert. Dieser generische Befehl steht auch dem Anwender zur Verfügung, um eigene Varianten – etwa mit einem verstärkten Bruchstrich oder mit anderen Klammern – zu setzen.

Um alle diese Varianten setzen zu können, werden allerdings vier zusätzliche Argumente benötigt. Der Befehl hat daher die folgende Form:

```
\genfrac{<Lbg>}{<Rbg>}{<Stärke>}{<Mstyle>}
        {<Oben>}{<Unten>}
```

Die beiden ersten Argumente `<Lbg>` und `<Rbg>` sind linker und rechter Begrenzer, im Fall eines Binomialkoeffizienten also die öffnende und schließende runde Klammer. Erlaubt sind alle (einzelnen!) Klammersymbole. Zu beachten ist, dass stets beide Klammersymbole fehlen oder angegeben sein müssen. Soll nur einer der Begrenzer gesetzt werden, kann der andere wie bei `\left` oder `\right` als `.` (Punkt) angegeben werden. Eine Kombination aus `\big . . .` und einem Klammersymbol ist für die Argumente `<Lbg>/<Rbg>` nicht zulässig und führt zu einem Fehler. Die Klammersymbole sind in ihrer Größe leider auf die bei den Binomialkoeffizienten gezeigten Größen beschränkt.

Das dritte Argument `<Stärke>` ist eine *Längenangabe* und legt die Stärke des Bruchstrichs fest. Bei fehlender Angabe wird die Stärke eines normalen Bruchstrichs eingesetzt. Durch Angabe eines positiven Werts, beispielsweise `1.3pt`, kann ein verstärkter Bruchstrich erzeugt werden, um bei einem Mehrfachbruch den Hauptbruchstrich hervorzuheben und den Bruch damit optisch besser zu gliedern.

Das an vierter Stelle angegebene Argument `<Mstyle>` ist eine ganze Zahl zwischen 0 und 3 und legt die Darstellung (mathematischen Modus) fest, in dem das ‚Objekt‘ gesetzt werden soll. Zwischen den Zahlen und

dem Befehl zur Wahl des mathematischen Modus besteht dabei folgende Beziehung:

0	<code>\displaystyle</code>	wie abgesetzte Formeln,
1	<code>\textstyle</code>	wie Formeln im Fließtext,
2	<code>\scriptstyle</code>	wie Exponenten und Indizes,
3	<code>\scriptscriptstyle</code>	wie doppelt gestufte Exponenten/Indizes.

Bei fehlender Angabe erfolgt der Satz wie in abgesetzten Formeln.

Die beiden letzten Argumente sind die oben und unten stehenden Teilformeln wie Zähler und Nenner bei einem Bruch, also die üblichen Argumente.

Aufgrund der Anzahl der vorgeschriebenen Argumente eignet sich dieser Befehl nur bedingt zum Setzen Bruch-artiger Formelteile. Er ist allerdings hervorragend geeignet, um in einem `\newcommand`-Befehl abweichende Formen wie Brüche mit verstärkten Bruchstrichen zu definieren. Das empfiehlt sich spätestens dann, wenn ein `\genfrac`-Befehl mehrfach mit den gleichen ersten vier Argumenten verwendet wird.

Der `\genfrac`-Befehl bietet alle Möglichkeiten der `plainTeX`-Befehle `\over`, `\overwithdelims`, `\atop`, `\atopwithdelims`, `\above` sowie `\abovewithdelims`. Diese Befehle, die nicht der `LaTeX`-Syntax entsprechen, sollten nicht zusammen mit dem Paket `amsmath` verwendet werden und erzeugen entsprechende Warnungen.

Die Verwendung des `\genfrac`-Befehls lässt sich gut an Definitionen der – in ähnlicher Weise realisierten – Befehle `\frac`, `\binom` und `\tbinom` zeigen:

```
\newcommand{\frac}[2]{\genfrac{}{}{}{}{#1}{#2}}
\newcommand{\binom}[2]{\genfrac{(}{)}{0pt}{}{#1}{#2}}
\newcommand{\tbinom}[2]{\genfrac{(}{)}{0pt}
{1}{#1}{#2}}
```

Die erste Zeile zeigt, dass die Standardwerte für fehlende Argumente so festgelegt sind, dass die Ausgabe des normalen `\frac` erzeugt wird. Bei den Binomialkoeffizienten werden die beiden ersten Argumente für die öffnende (linke) und schließende (rechte) runde Klammer verwendet, und das dritte Argument setzt die Stärke des Bruchstrichs auf 0pt und verhindert damit seine Ausgabe. Die fehlende Angabe im vierten Argument bei `\frac` und `\binom` führt dazu, dass die Ausgabe automatisch an den aktuellen mathematischen Modus angepasst wird; der Wert 1 an dieser Stelle bei `\tbinom` erzwingt eine Ausgabe wie bei Inline-Formeln.

Die folgenden Beispiele zeigen unterschiedliche Ausgaben des `\genfrac`-Befehls. Bei mehrfacher Nutzung in einem Text empfiehlt es sich, einen

eigenen neuen Befehl wie oben gezeigt zu definieren und verwenden. Das ist hier für den Fall eines Bruchs mit verstärktem Bruchstrich realisiert.

```

9  \newcommand{\Bruch}[2]{%
10   \genfrac{}{}{1.3pt}{}%
11   {#1}{#2}}
12  \begin{gather*}
13   \frac{\dfrac{x+y}{x^2+y^2}}{1+\dfrac{a-b}{x+\dfrac{a+b}{(a-b)^2}}}
14   \frac{x+y}{x^2+y^2}
15   %
16   %
17   \Bruch{\dfrac{x+y}{x^2+y^2}}{1+\dfrac{a-b}{x+\dfrac{a+b}{(a-b)^2}}}
18   \frac{x+y}{x^2+y^2}
19   %
20  \end{gather*}
21  \end{document}

```

$$\frac{\frac{x+y}{x^2+y^2}}{1+\frac{a-b}{x+\frac{a+b}{(a-b)^2}}}$$

[Buch1/AmSPakete/genfrac.tex]

Das zweite Beispiel zeigt, wie mittels des `\genfrac`-Befehls doch eine Anpassung der Klammerngröße möglich ist. Die Begrenzer werden nicht beim `\genfrac`-Befehl angegeben, sondern schließen als angepasste Klammern den `\genfrac`-Befehl ein.

```

9  \begin{gather*}
10   \left(\genfrac{}{}{0pt}{}{\dfrac{x+y}{x^2+y^2}}{1+\dfrac{a-b}{x+\dfrac{a+b}{(a-b)^2}}}\right)
11   \left[\medskipamount\right]
12   \left.\genfrac{}{}{0pt}{}{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\dbinom{n+k}{m}}}\right\}
13   \left.\genfrac{}{}{0pt}{}{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\dbinom{n+k}{m}}}\right\}
14   \left.\genfrac{}{}{0pt}{}{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\dbinom{n+k}{m}}}\right\}
15   \left.\genfrac{}{}{0pt}{}{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\dbinom{n+k}{m}}}\right\}
16   \left.\genfrac{}{}{0pt}{}{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\dbinom{n+k}{m}}}\right\}
17   \left.\genfrac{}{}{0pt}{}{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\dbinom{n+k}{m}}}\right\}
18   \left.\genfrac{}{}{0pt}{}{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\dbinom{n+k}{m}}}\right\}
19   \left.\genfrac{}{}{0pt}{}{\dbinom{n+k}{m+k}}{1+\dbinom{n}{k+\dbinom{n+k}{m}}}\right\}

```

$$\left(\frac{\frac{x+y}{x^2+y^2}}{1+\frac{a-b}{x+\frac{a+b}{(a-b)^2}}} \right)$$

[Buch1/AmSPakete/genfrac2.tex]

12.1.5.7 Integrale

amsmath stellt auch einige zusätzliche Symbole für Mehrfachintegrale zur Verfügung:

```
\iint
\iiint
\iiiiint
\idotsint
```

In Standard- $\text{\LaTeX}$ setzt der Befehl `\int` ein einfaches Integral. Jedes weitere `i` steht für ein weiteres Integralzeichen. Also ist `\iint` das Symbol für ein Doppel-, `\iiint` für ein Dreifach- und `\iiiiint` für ein Vierfachintegral. Für allgemeine Mehrfachintegrale gibt es das Symbol `\idotsint`.

Die Symbole selbst zeigt das folgende Beispiel, in dem jedes der Symbole für Mehrfachintegrale verwendet wird.

```
9 \begin{gather*}
10 \iint\limits_A
11 f(x,y)\,:\mathrm{d}x\,:\mathrm{d}y\,
12 \iiint\limits_A
13 f(x,y,z)\,:\mathrm{d}x\,:\mathrm{d}y\,:\mathrm{d}z\,
14 \iiiiint\limits_A
15 f(w,x,y,z)\,:\mathrm{d}w\,:\mathrm{d}x\,:\mathrm{d}y\,:\mathrm{d}z\,
16 \idotsint\limits_A
17 f(x_1,\,\ldots,\,x_k)
18 \end{gather*}
```

$$\begin{aligned} & \iint_A f(x,y) \, dx \, dy \\ & \iiint_A f(x,y,z) \, dx \, dy \, dz \\ & \iiiiint_A f(w,x,y,z) \, dw \, dx \, dy \, dz \\ & \int \cdots \int_A f(x_1, \dots, x_k) \end{aligned}$$

[Buch1/AmSPakete/iiint.tex]

Der `\limits`-Befehl wird genutzt, um die Menge A , über die integriert wird, unter das Integralsymbol zu setzen. Der Indizierungsbefehl ohne diesen Zusatz würde das A neben das jeweilige Integral setzen, wie unten bei B zu sehen.

Vor dem Differential d muss der Abstand korrigiert werden, da dieser Buchstabe keine normale Variable darstellt (vgl. 12.1.5.17). Alternativ oder zusätzlich kann es geradestehend gesetzt werden. Die Art, wie ein Differential gesetzt wird, variiert bei verschiedenen Publikationen.

```
9 \begin{gather*}
10 \iint_B
11 f(x,y)\,,\,\mathrm{d}x\,
12 \,,\,\mathrm{d}y\,
13 \end{gather*}
14 \end{document}
```

$$\iint_B f(x,y) \, dx \, dy$$

[Buch1/AmSPakete/iiintd.tex]

12.1.5.8 Modulofunktionen

$\text{\LaTeX}$ stellt zwei Modulofunktionen `\bmod` und `\pmod` zur Verfügung. `amsmath` modifiziert die beiden Befehle in der Weise, dass die Abstände bei nicht abgesetzten Formeln (etwas) verkürzt werden. Zusätzlich definiert `amsmath` die beiden Operatoren

```
\mod
\pod
```

als alternative Formen der beiden Befehle. Bei `\mod` fehlen die bei `\pmod` verwendeten Klammern, und `\pod` unterscheidet sich von `\pmod` durch das Fehlen des Texts ‚mod‘.

Das folgende Beispiel zeigt die vier Modulobefehle. Da es sich um abgesetzte Formeln handelt, wird die erste Formel ohne Unterschied zu Standard- $\text{\LaTeX}$ gesetzt.

```
9 \begin{align*}
10 & \gcd(n,m\bmod n)\\
11 & x\equiv y\pmod b\\
12 & x\equiv y\mod c\\
13 & x\equiv y\pod d\\
14 \end{align*}
```

$$\gcd(n, m \bmod n)$$

$$x \equiv y \pmod{b}$$

$$x \equiv y \mod c$$

$$x \equiv y \pod{d}$$

[Buch1/AmSPakete/mod.tex]

12.1.5.9 Umrahmte Formeln und Formelteile

Hervorhebungen von Formeln oder Formelteilen durch Schriftumschaltungen sind unüblich, da in vielen Fällen die für die Variablen verwendete Schrift schon eine Bedeutung hat. Insbesondere komplette Formeln hebt man statt dessen gerne durch einen Rahmen um die Formel hervor. `amsmath` stellt dafür den Befehl `\boxed` zur Verfügung, der um das angegebene Argument einen Rahmen zeichnet, ähnlich dem `\fbox`-Befehl im Textmodus. Statt auf die gesamte Formel kann der Befehl auch auf eine Teilformel angewendet werden.

```
\boxed{<Teilformel>}
```

Das einzige Argument des Befehls ist die Teilformel, um die ein Rahmen gesetzt werden soll. Da `\boxed` intern mit Hilfe eines `\fbox`-Befehls realisiert ist, können die Voreinstellungen der Strichstärke des Rahmens

und des Abstands von der Teilformel über die Längen `\fboxrule` und `\fboxsep` verändert werden. Rahmen um mehrzeilige Formeln lassen sich mit Hilfe der Umgebung `split` setzen.

Dazu ein Beispiel, das auch die Verwendung der erwähnten Längen zum Ändern der Voreinstellungen demonstriert.

```

9  \begin{gather*}
10 \boxed{\sin 2x =
11   2\sin x \cos x}\\
12 \setlength{\fboxrule}{%
13   {3\fboxrule}}
14 \setlength{\fboxsep}{3mm}
15 \boxed{\cos 2x =
16   \cos^2 x - \sin^2 x}\\
17 \boxed{\begin{split}
18   \sin x
19   &= \sqrt{1-\cos^2 x}\\
20   &= \frac{1}{\sqrt{1+\cot^2 x}}
21   +\cot^2 x}}
22 \end{split}}
23 \end{gather*}

```

[Buch1/AmSPakete/boxed.tex]

Wie man sieht, wirkt sich die Veränderung der Box nur auf den unmittelbar folgenden `\boxed`-Befehl aus. Möchte man alle Formeln eines gesamten Dokuments in dieser veränderten Art und Weise umrahmen, so müssen diese Modifikationen in die Präambel des Dokuments verschoben werden.

12.1.5.10 Unter- und Oberlängen von Teilformeln

Die Unter- und Oberlängen von Teilformeln können zu unerwünschten Formatierungen führen. Ein typisches Beispiel ist $\sqrt{x} + \sqrt{y}$, wo die Unterlänge des Buchstabens y zu einer in diesem Fall unerwünschten Verlängerung des zugehörigen Wurzelzeichens nach unten führt. In `amsmath` gibt es zur Korrektur den Befehl `\smash`, der die Unter- bzw. Oberlänge seines Arguments annulliert. Er hat die Form

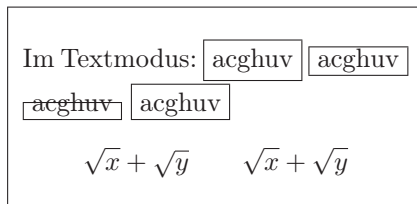
```
\smash[b | t]{<Argument>}
```

und kann auch im Textmodus eingesetzt werden. Bei fehlendem optionalem Argument wird `tb` angenommen, wobei `b` die Unter- und `t` die Oberlänge von `<Argument>` unterdrückt.

Die Wirkung des Befehls in beiden Modi zeigen die folgenden Beispiele.

```

9   Im Textmodus: \fbox{acghuv}
10  \fbox{ac\smash{gh}uv}
11  \fbox{\smash{acghuv}}
12  \fbox{\smash[b]{acghuv}}
13  \[
14  \sqrt{x}+\sqrt{y} \quad \qquad
15  \sqrt{x}+\sqrt{\smash[b]{y}}
16  \]
```



[Buch1/AmSPakete/smash.tex]

Der Textmodus im Beispiel zeigt, dass das Unterdrücken von Oberlängen leicht zu einer fehlerhaften Formatierung führen kann. In den meisten Fällen wird daher nur die Unterlänge durch Angabe des optionalen Arguments `b` unterdrückt.

12.1.5.11 Spezielle Pfeile

$\text{\LaTeX}$ stellt die Befehl `\overleftarrow` und `\overrightarrow` zur Verfügung, um links- oder rechtsgerichtete Pfeile über Teilformeln zu setzen. `amsmath` definiert zusätzlich Pfeile, die *unter* statt über die Teilformel gesetzt werden:

```

\underleftarrow{<Teilformel>}
\underrightarrow{<Teilformel>}
```

Die Befehl haben die gleiche Form wie die von $\text{\LaTeX}$ definierten Befehle. Auch Pfeile in beide Richtungen gleichzeitig können über und unter Formeln gesetzt werden. Die Befehl lauten analog:

```

\overleftrightharrow{<Teilformel>}
\underleftrightharrow{<Teilformel>}
```

Weitere Pfeilformen dienen beispielsweise der Definition von Abbildungen und passen sich in der Länge an. Sie werden in der Verarbeitung als Operatoren behandelt und haben die folgende Form:

```

\xleftarrow[<Untere TF>]{<Obere TF>}
\xrightarrow[<Untere TF>]{<Obere TF>}
```

Ihre Länge wird bestimmt durch die längere der beiden Teilformeln (*<Untere TF>*) und (*<Obere TF>*). Eine obere Teilformel muss angegeben werden, kann aber leer sein. Die Angabe einer unteren Teilformel ist optional, wie ja die Argumentschreibweise in eckigen Klammern signalisiert.

Das folgende Beispiel zeigt die Anwendung der sechs Pfeilformen über und unter Teilformeln und die beiden Abbildungsdefinitions Pfeile.

```

9  \[
10 \overleftrightharpoonup{f \circ h}
11 \overleftarrow{f \circ g}\qquad
12 \overrightarrow{g \circ h}\qquad
13 \] \[
14 \underleftrightharpoonup%
15   {f \circ h}
16   \qquad
17 \underleftarrow{f \circ g}
18   \qquad
19 \underrightarrow{g \circ h}
20 \] \[
21 A \xrightarrow{f \circ g} B
22   \qquad
23 B \xrightarrow[g \circ h]{} C
24 \] \[
25 B \xleftarrow{f \circ g} A
26   \qquad
27 A \xrightarrow[F]{f \circ h \circ h} C
28 \]
```

$$\begin{array}{ccc}
 \overleftrightharpoonup{f \circ h} & \overleftarrow{f \circ g} & \overrightarrow{g \circ h} \\
 \underleftrightharpoonup{f \circ h} & \underleftarrow{f \circ g} & \underrightarrow{g \circ h} \\
 A \xrightarrow{f \circ g} B & B \xrightarrow[g \circ h]{} C & \\
 B \xleftarrow{f \circ g} A & A \xrightarrow[F]{f \circ h \circ h} C &
 \end{array}$$

[Buch1/AmSPakete/arrows.tex]

12.1.5.12 Kombinieren von Symbolen

In der Mathematik setzt man gelegentlich Symbole oder Teilformeln über oder unter andere Symbole oder Teilformeln. $\LaTeX$ hat dafür den Befehl `\stackrel`, der eine Teilformel in der Schriftgröße eines Exponenten über eine Teilformel in der aktuellen Schriftgröße setzt. Ein Befehl zum setzen einer Teilformel in kleinerer Schriftgröße unter eine andere Teilformel ist nicht vorgesehen.

Das Paket amsmath definiert Befehle für beide Varianten.

```

\overset{<Kleine TF oben>}{<Teilformel>}
\underiset{<Kleine TF unten>}{<Teilformel>}

```

Das erste Argument ist eine Teilformel, die in der Schriftgröße eines Exponenten von `\overset` über und von `\underset` unter die als zweites Argument angegebene Teilformel gesetzt wird. Im Beispiel am Ende dieses Abschnitts werden auch diese Befehle verwendet.

Exponenten und Indizes setzt $\text{\LaTeX}$ stets rechts vom Symbol. In mathematischen Formeln werden gelegentlich aber auch links von einem Symbol oder an allen vier Ecken derartige Teilformeln angebracht. `amsmath` bietet einen Befehl an, der eine Teilformel um zusätzliche (kurze) Teilformeln links und rechts von ihr ergänzt. In der Regel werden die linken und rechten Teilformeln lediglich aus Exponenten oder Indizes bestehen oder allenfalls aus einzelnen Sonderzeichen wie einem einfachen Anführungszeichen, dem Befehl `\prime` oder ähnlichen Zeichen. Der Befehl lautet

```
\sideset{<Links>}{<Rechts>}{<Mittlere TF>}
```

mit den linken und rechten Teilformeln `<Links>` und `<Rechts>`. An die mittig gesetzte `<Mittlere TF>` kann ein `\nolimits`-Befehl angehängt werden – etwa eine Funktion oder ein ‚großes‘ Symbol. Meist handelt es sich ja bei der Teilformel um ein einzelnes ‚großes‘ Symbol wie `\sum` oder `\prod`. Falls gewünscht lässt sich allerdings jede beliebige Teilformel mit dem Befehl `\operatorname` (vgl. 12.2.3, S. 424) zu einer Funktion machen, die als mittlere Teilformel angegeben werden kann. Alle Teilformeln werden in der aktuellen Größe gesetzt, wobei Exponenten und Indizes wie üblich behandelt werden. Das Beispiel am Ende zeigt auch diesen Befehl.

Bei Summen beispielsweise sind mehrzeilige Begrenzungen ein Problem. Im Standard- $\text{\LaTeX}$ lässt sich dieses Problem lösen mit der Umgebung `array` und manueller Wahl der Schriftgröße. `amsmath` stellt für mehrzeilige Exponenten und Indizes den einfachen Befehl

```
\substack{<Zeile 1>[ \<Zeile 2> ]...}
```

zur Verfügung, der in jedem Exponenten oder Index verwendet werden kann. Die durch `\&` getrennten Zeilen werden zentriert gesetzt. Zeilen- und andere Abstände sind deutlich kleiner als bei einem `array` und damit besser an diese spezielle Verwendung angepasst.

Eine Variante des `\substack`-Befehls ist die Umgebung `subarray`. Der `\substack`-Befehl basiert auf dieser Umgebung. Es handelt sich dabei um ein spezielles `array`, das auf eine einzige Spalte beschränkt ist. Einzige sinnvolle – trotzdem aber erforderliche – Spaltenspezifikation ist ein einzelnes `l`, also eine linksbündige Spalte:

```
\begin{subarray}{l}
  <Zeile 1> |\\
  <Zeile 2> |...
\end{subarray}
```

Grundsätzlich ist auch die Spaltenangabe `c` möglich, die der (kürzere) `\substack`-Befehl verwendet. Die Abstände entsprechen denen des `\substack`-Befehls.

Das abschließende Beispiel zeigt die gerade behandelten Befehle und die Umgebung. Die erste Formel in den Zeilen 10–12 zeigt Anwendungen von `\overset` und `\underset`. Die beiden Formeln in den Zeilen 15–19 und 21–23 zeigen den `\sideset`-Befehl. In der letzter Formel, Zeilen 25–31, werden `\substack` und `subarray` angewendet.

```
9  \[
10 \overset{\rightarrow}{X} \cdot
11 \underset{\leftarrow}{W}
12 \underset{-}{*}
13 \overset{a+b}{a+\cdots+h}
14 \] \[
15 \sum_{n=1}^{\infty} E_n \quad
16 \sideset{}{'}{\sum}_{n<k, \text{odd}} H_n \quad
17 \sideset{}{^+}{\prod}_{n=1}^{\infty} \text{prime}
18 \] \[
19 \sideset{}{^*}{\sin} \cos^* \sideset{}{^+}{A}_x
20 \quad \quad \quad \sum_{0 \leq i < m} a_{ij} \prod_{\substack{k \in K \text{ odd} \\ l \in L \text{ even}}} f_k \circ g_l
21 \] \[
22 \sum_{0 \leq j < n} a_{ij} \quad
23 \prod_{\substack{k \in K \text{ odd} \\ l \in L \text{ even}}} f_k \circ g_l
24 \] \[
25 \sum_{\substack{0 \leq i < m \\ 0 \leq j < n}} a_{ij} \quad
26 \prod_{\substack{k \in K \text{ odd} \\ l \in L \text{ even}}} f_k \circ g_l
27 \] \[
28 \sum_{\substack{k \in K \text{ odd} \\ l \in L \text{ even}}} f_k \circ g_l
29 \] \[
30 \sum_{\substack{k \in K \text{ odd} \\ l \in L \text{ even}}} f_k \circ g_l
31 \] \[
32 \] \]
```

$$\begin{aligned} & \overset{a+b}{\underset{\leftarrow}{X}} \cdot \underset{-}{W} \underset{*}{a} + \cdots + h \\ & \sum' E_n \quad \sum'_{n < k, n \text{ odd}} H_n \quad \prod'_{n=1}^{\infty} \text{prime} \\ & \quad \quad \quad \sin^* \cos^* \quad \sideset{}{^+}{A}_x \\ & \sum_{\substack{0 \leq i < m \\ 0 \leq j < n}} a_{ij} \quad \prod_{\substack{k \in K \text{ odd} \\ l \in L \text{ even}}} f_k \circ g_l \end{aligned}$$

[Buch1/AmSPakete/scomb.tex]

12.1.5.13 Klammern

$\text{\LaTeX}$ verfügt über Klammern unterschiedlicher Größe. Die Größe kann vom Anwender explizit vorgegeben werden, indem er den zusammengehörenden linken und rechten Klammersymbolen wie bei der Behandlung

des mathematischen Modus beschrieben eine der Größenangaben `\bigl` und `\bigr`, `\biggl` und `\biggr`, `\Bigl` und `\Bigr` sowie `\Biggl` und `\Biggr` voranstellt. $\text{\LaTeX}$ skaliert die Klammersnößen allerdings nicht korrekt über sämtliche verwendeten Fontgrößen. `amsmath` behebt dieses Problem und skaliert die Größe der ‚großen‘ Klammern korrekt über alle von $\text{\LaTeX}$ verwendeten Schriftgrößen.

Eine automatische Anpassung nehmen die Größenangaben `\left` und `\right` vor, wenn sie vor den zusammengehörenden linken und rechten Klammersymbolen stehen. Mehrfache mit `\left` ausgezeichnete Klammern links von einer Teilformel mit zugehörigem `\right` rechts haben allerdings alle die gleiche Größe.

Beispiele zu den Klammern sind im Kapitel über den Formelsatz zu finden. Dort werden auch die Vor- und Nachteile der beiden Methoden behandelt (vgl. 6, S. 137).

Ergänzend zu den von $\text{\LaTeX}$ angebotenen Klammersymbolen stellt `amsmath` die Klammern

```
\lvert
\rvert
\lVert
\rVert
```

zur Verfügung. `\lvert` und `\rvert` sind Synonyme für das Klammersymbol `|`. Die Befehle `\lVert` und `\rVert` sind alternative Namen für die Klammer `|`. Anlass für diese neuen Befehle ist, dass der senkrechte Strich in der Mathematik und im mathematischen Modus mit einer ganzen Reihe von unterschiedlichen Bedeutungen verwendet wird. Beispielsweise in $p|q$ für die Relation ‚Teiler‘ der Zahlentheorie, in $|x|$ für den Absolutbetrag oder in $f_n(t)|_{t=0}$ für eine Funktion ausgewertet an der Stelle $t = 0$.

Das Problem sind unterschiedliche typografische Konventionen für die verschiedenen Bedeutungen, die ein Programm nicht mit der gleichen Sicherheit unterscheiden kann wie ein damit vertrauter Autor oder Leser. Daher sollte ein Autor unterschiedliche Befehle für die bei ihm auftretenden Bedeutungen definieren. Da linke und rechte Klammern von $\text{\LaTeX}$ hinsichtlich der benachbarten Symbole unterschiedlich behandelt werden, ist es vorteilhaft, linke und rechte Symbole ebenfalls zu unterscheiden. Dies gilt auch, wenn für links und rechts das gleiche Klammersymbol verwendet wird. Damit könnte ein Autor die Befehle

```
\newcommand{\abs}[1]{\lvert#1\rvert}
\newcommand{\norm}[1]{\lVert#1\rVert}
```

definieren, um mit `\(\abs{x}\)` den Absolutbetrag $|x|$ und mit `\(\norm{f}\)` die Norm $\|f\|$ zu setzen.

12.1.5.14 Akzente

$\text{\LaTeX}$ positioniert den zweiten Akzent bei doppelten mathematischen Akzenten häufig nicht optimal. Mit amsmath ist diese Platzierung besser. Zusätzlich zu den Standardakzenten `\dot` und `\ddot` definiert amsmath die Akzente

```
\dddot{<Zeichen>}
\ddddot{<Zeichen>}
```

für Akzente, die aus drei und vier nebeneinander liegenden Punkten bestehen.

Um beliebige Symbole in der Position eines mathematischen Akzents oder eines Akzents unter einem Symbol zu setzen, verweist das Benutzerhandbuch zu amsmath auf das Paket accents von Javier Bezos, das allerdings erst in einem späteren Band dieses Werks behandelt wird.

12.1.5.15 Folgepunkte

$\text{\LaTeX}$ kennt für Folgepunkte auf der Grundlinie den Befehl `\ldots` und für solche auf der Höhe von Operatoren den Befehl `\cdots`. Diese Befehle legen die Höhe der Folgepunkte explizit fest und müssen daher ausgetauscht werden, wenn sich die zu Grunde liegenden Konventionen ändern. Das Paket amsmath sieht an ihrer Stelle fünf Befehle vor, die sich an der Bedeutung der Folgepunkte orientieren und entweder als `\cdots` oder als `\ldots` gesetzt werden. Neue Varianten der Folgepunkte sind also nicht verfügbar. Der Vorteil der fünf Befehle liegt vielmehr darin, dass sie sich leicht unterschiedlichen Konventionen anpassen lassen, ohne das gesamte Dokument zu bearbeiten. Vorgesehen sind die in der folgenden Liste beschriebenen Befehle.

`\dotsb` Folgepunkte für binäre Operatoren und Funktionen
(entspricht `\cdots`),

`\dotsc` Folgepunkte zwischen Kommas
(entspricht `\ldots`),

`\dotsi` Folgepunkte für Integralzeichen
(entspricht `\cdots`),

`\dotsm` Folgebunkte für die Multiplikation
(entspricht `\cdots`),

`\dotso` Folgebunkte für andere als die zuvor genannten Situationen
(entspricht `\ldots`).

Gegebenenfalls korrigieren diese Befehle auch die Abstände vor und nach den Folgebunkten. Die Voreinstellungen sind entsprechend den Konventionen der $\mathcal{AMS}$ gewählt.

Die Verwendung der Befehle in der Voreinstellung:

```

9  \[ \int_{A_1}\dotsm\int_{A_n}
10    f(a_1), \dotsc f(a_n) \]
11  \[ x_1 + x_2 + \dotsb + x_n,
12    \quad t_1 t_2 \dotsm \]
13  \[ \sum_{i_1 \in I_1} \dotso
14    \sum_{i_n \in I_n} \]
```

$$\int_{A_1} \cdots \int_{A_n} f(a_1), \dots f(a_n)$$

$$x_1 + x_2 + \cdots + x_n, \quad t_1 t_2 \cdots$$

$$\sum_{i_1 \in I_1} \cdots \sum_{i_n \in I_n}$$

[Buch1/AmSPakete/dots.tex]

12.1.5.16 Varianten großer griechischer Buchstaben

Für die großen griechischen Buchstaben, die nicht durch lateinische Buchstaben dargestellt werden, stellt `amsmath` kursive Varianten zur Verfügung. Tabelle 12.1 zeigt die Buchstaben zusammen mit den Befehlen, durch die sie gesetzt werden.

Tabelle 12.1: Varianten großer griechischer Buchstaben im Italic-Font

Γ <code>\varGamma</code>	Ξ <code>\varXi</code>	Φ <code>\varPhi</code>
Δ <code>\varDelta</code>	Π <code>\varPi</code>	Ψ <code>\varPsi</code>
Θ <code>\varTheta</code>	Σ <code>\varSigma</code>	Ω <code>\varOmega</code>
Λ <code>\varLambda</code>	Υ <code>\varUpsilon</code>	

12.1.5.17 Feinpositionierung und Abstände

Gerade für den mathematischen Satz ist die automatische Erkennung der Bedeutung eines Symbols und damit des benötigten Abstands schon tief in $\text{T}_{\text{E}}\text{X}$ verankert. $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ und die Pakete der $\mathcal{AMS}$ haben dies übernommen und verfeinert. Dennoch ist es manchmal notwendig, manuell

eingzugreifen und an wenigen Stellen kleine Korrekturen vorzunehmen. Ein Neuling im Buchsatz sollte diese Feinheiten zunächst beiseite lassen und sich erst später damit beschäftigen. Geübtere Anwender sind sicher in der Lage, beispielsweise ein gerade stehendes Differential (s. 408 von vorne herein als eigenen Befehl zu definieren. Dennoch ist davor zu warnen, Abstandskorrekturen ad hoc vorzunehmen, die Praxis zeigt, dass dies meist zu unsystematischen oder für den Leser gar irritierenden Abständen führt.

$\text{\LaTeX}$ definiert zum Gebrauch im mathematischen Modus die Abstände `\,` bzw. `\thinspace`, `\:`, `\;`, `\quad`, `\qquad` und den negativen Abstand `\!` bzw. `\negthinspace`. Das Paket `amsmath` definiert zusätzliche Namen für die Befehle `\:` und `\;` sowie Befehle für die zugehörigen negativen Abstände. Zusätzlich stellt es einen Befehl `\mspace` zur Verfügung, der dem `\hspace`-Befehl entspricht. Die folgenden neuen Abstandsbeefehle kommen damit zu den bekannten hinzu:

```
\medspace
\thickspace
\negmedspace
\medthickspace
\mspace{Länge}
```

Als Längeneinheit beim `\mspace`-Befehl wird zur Feinpositionierung der mathematischen Abstände die Längeneinheit

$$1\text{ mu} = \frac{1}{18}\text{ em}$$

des mathematischen Modus empfohlen. Ein `em` ist die Breite des Buchstabens „M“ im aktuellen Zeichensatz.

Die Bedeutung der Abstände zeigt am Besten der folgende tabellarische Vergleich:

Befehl(e)	Beispiel	Befehl(e)	Beispiel
ohne Abstandsbeefehl	$\Rightarrow\Leftarrow$	ohne Abstandsbeefehl	$\Rightarrow\Leftarrow$
<code>\,</code> <code>\thinspace</code>	$\Rightarrow\Leftarrow$	<code>\!</code> <code>\negthinspace</code>	$\Rightarrow\Leftarrow$
<code>\:</code> <code>\medspace</code>	$\Rightarrow\Leftarrow$	<code>\negmedspace</code>	$\Rightarrow\Leftarrow$
<code>\;</code> <code>\thickspace</code>	$\Rightarrow\Leftarrow$	<code>\negthickspace</code>	$\Rightarrow\Leftarrow$
<code>\mspace{8mu}</code>	$\Rightarrow\Leftarrow$	<code>\mspace{-8mu}</code>	$\Rightarrow\Leftarrow$
<code>\quad</code>	$\Rightarrow\Leftarrow$		
<code>\qquad</code>	$\Rightarrow\Leftarrow$		

12.2 Weitere $\mathcal{AMS}$ -Pakete

12.2.1 Kommutative Diagramme mit `amscd`

Zum Erstellen einfacher kommutativer Diagramme kann das Paket `amscd` verwendet werden. Für komplexe kommutative Diagramme wird in der Dokumentation zu `amsmath` auf die Pakete `kuvio` oder `xypic` verwiesen.

Das Paket wird nicht von `amsmath` geladen. Sollen mit ihm kommutative Diagramme erstellt werden, muss das Paket explizit mit

```
\usepackage{amscd}
```

geladen werden.

Mit `amscd` erstellte kommutative Diagramme können nur horizontale und vertikale Pfeile enthalten. Im mathematischen Modus können sie mit Hilfe der Umgebung `CD` gesetzt werden. Aufgrund der vertikalen Ausdehnung kommutativer Diagramme, die in der Regel aus drei oder mehr Zeilen bestehen, sollten sie nach Möglichkeit nicht in Inline-Formeln verwendet werden. Die Umgebung wird im mathematischen Modus wie folgt aufgerufen:

```
\begin{CD}
  <Erste Zeile> [ \\\
  <Weitere Zeile(n)> ] ...
\end{CD}
```

Die einzelnen Zeilen bestehen entweder aus Teilformeln, getrennt durch horizontale Pfeile, oder sie beginnen mit einem vertikalen Pfeil, dem weitere Pfeile folgen. Im zweiten Fall können Teilformeln nur zur Beschriftung der Pfeile angegeben werden.

Pfeile werden in einer speziellen Form angegeben und haben nicht die Form von Befehlen. Statt dessen wird jedes Pfeilsymbol mit dem Zeichen `@` eingeleitet. Die folgenden ‚Pfeile‘ stehen zur Verfügung:

<code>@>[<oben>]>[<unten>]></code>	Horizontaler Pfeil nach rechts
<code>@<[<oben>]<[<unten>]<</code>	Horizontaler Pfeil nach links
<code>@V[<links>]V[<rechts>]V</code>	Horizontaler Pfeil nach unten
<code>@A[<links>]A[<rechts>]A</code>	Horizontaler Pfeil nach oben
<code>@=</code>	Horizontale Doppellinie
<code>@ </code>	Vertikale Doppellinie
<code>@.</code>	‚Unsichtbarer‘ Pfeil

Die eigentlichen Pfeile sind durch drei gleiche Symbole gekennzeichnet, die die Richtung des Pfeils festlegen. Die optionalen Teilformeln `<oben>`, `<unten>`, `<links>` und `<rechts>` dienen zur Beschriftung des Pfeils an der ihrem Namen entsprechenden Position. Für die Doppellinien und den ‘unsichtbaren’ Pfeil ist keine Beschriftung vorgesehen.

Die folgenden Beispiele zeigen mit der `CD`-Umgebung gesetzte kommutative Diagramme.

```

9  \newcommand{\PP}{\prime\prime}
10 \begin{gather*}
11 \begin{CD}
12 X @>h_X>> X^{\PP} \\
13 @VVV @VVV \\
14 Y @>h_Y>> Y^{\PP}
15 \end{CD} \\
16 %
17 \begin{CD}
18 S^{\mathcal{W}_\Lambda} \otimes T @>j>> T \\
19 @VVV @VVV \\
20 (S \otimes T)/I @= (Z \otimes T)/J
21 \end{CD}
22 \end{gather*}
23
24 \end{gather*}
25 \end{gather*}

```

$$\begin{array}{ccc}
 X & \xrightarrow{h_X} & X'' \\
 \downarrow F & & \downarrow F'' \\
 Y & \xrightarrow{h_Y} & Y''
 \end{array}$$

$$\begin{array}{ccc}
 S^{\mathcal{W}_\Lambda} \otimes T & \xrightarrow{j} & T \\
 \downarrow & & \downarrow \text{End } P \\
 (S \otimes T)/I & = & (Z \otimes T)/J
 \end{array}$$

[Buch1/AmSPakete/amscd.tex]

12.2.2 Paket `amsbsy`

Dieses kleine Paket ermöglicht das Setzen fetter Symbole (engl. *bold symbols*) im mathematischen Modus. Es wird geladen mit

```
\usepackage{amsbsy}
```

und besitzt keine Optionen. `amsbsy` verwendet das Hilfspaket `amsgen` aus der Sammlung `amsmath`.

Das Paket definiert zwei Befehle, um Zeichen im mathematischen Modus fett zu setzen. Der Befehl

```
\boldsymbol{<Teilformel>}
```

setzt sein Argument in der Schriftversion *bold* (`\mathversion{bold}`). Zeichen erscheinen allerdings nur dann fett, wenn es einen fetten (mathematischen) Font gibt, der sie enthält. Fehlt ein fatter Font, wird das Zeichen in der normalen Strichstärke gesetzt. `\boldsymbol` ist in der Lage, Zahlen, Buchstaben aus dem Font `math italic`, griechische Buchstaben und ein Reihe von anderen Symbolen fett zu setzen, für die fette Fonts verfügbar sind.

Für Zeichen, zu denen eine fette Version fehlt, kann der Befehl

```
\pmb{<Teilformel>}
```

Fettdruck simulieren, indem er sein Argument leicht verschoben mehrfach druckt. Eine Korrektur der Abstände der Zeichen wird nicht vorgenommen, so dass die Zeichen enger gesetzt werden als bei ‚echtem‘ Fettdruck, und das Überdrucken der Zeichen geht natürlich zu Lasten der Qualität. Qualitätseinbußen müssen vor allem bei Symbolen mit Serifen oder spitz zulaufenden Enden wie beispielsweise runden Klammern in Kauf genommen werden (bei entsprechender Vergrößerung sind etwa die geteilten Enden der runden Klammern zu erkennen). Das Setzen fetter Symbole mit `\pmb` sollte daher nur als letzte Möglichkeit in Betracht gezogen werden, wenn eine fette Version von Zeichen nicht verfügbar ist. Das Argument von `\pmb` sollte auch nur solche Zeichen enthalten. Wann immer möglich, sollte statt dessen der `\boldsymbol`-Befehl verwendet werden.

Das folgende Beispiel zeigt die Verwendung dieser beiden Befehle. Die Formel in den Zeilen 7–12 zeigt einige Symbole in normaler Schrift (Zeile 8), in mit `\boldsymbol` gesetzter Fettschrift (Zeile 9) und in Zeile 10 in mit `\pmb` simulierter Fettschrift nebeneinander.

In den beiden Formeln in den Zeilen 11–19 wird eine Summe in verschiedenen Varianten gesetzt. Die erste Formel zeigt sie zunächst in der normalen Schrift gesetzt (Zeile 12) und dann in echter Fettschrift (Zeilen 13, 14). Das Summensymbol aus Zeile 13 erscheint dabei nicht fett, da es zu den ‚großen Symbolen‘ gehört, die es nicht in einer fetten Version gibt. In Zeile 16 und 17 wird daher das Summensymbol mit den Grenzen mit Hilfe von `\pmb` gesetzt und der Rest der Formel mit `\boldsymbol`. Der Summationsbereich (als Index oder Exponent gesetzt) muss in die emulierte Fettschrift einbezogen werden, da `\pmb{\sum}` für $\text{\LaTeX}$ kein mathematischer Operator ist und daher der `\limits`-Befehl nicht erlaubt ist. Zeile 18 setzt die gesamte Summe mit `\pmb`.

Die Formeln aus den Zeilen 19–27 zeigen ein Integral in normaler Schrift, in echter Fettschrift und in emulierter Fettschrift. In den Zeilen 21–24

könnte statt des `\boldsymbol`-Befehls in der Formel zwischen dem Ende der vorhergehenden Formel und dem Beginn der aktuellen Formel in Zeile 21 der Befehl `\boldmath` oder `\mthversion{bold}` verwendet werden, da die komplette Formel fett gesetzt wird. Zu beachten ist allerdings, dass beide Befehle auch für die folgenden Formeln gelten würden, wenn sie nicht gemeinsam mit der Formel in geschweifte Klammern eingeschlossen werden.

```

7  \[
8  \pm aB \beta \Gamma \quad \pm aB \beta \Xi \quad \pm aB \beta \Xi
9  \boldsymbol{\pm aB \beta \Xi}
10 \quad \pmb{\pm aB \beta \Xi}
11 \] \[
12 \sum_{x \in X} \ln x \quad \sum_{x \in X} \ln x
13 \boldsymbol{\sum_{x \in X} \ln x}
14 \quad \pmb{\sum_{x \in X} \ln x}
15 \] \[
16 \int_a^b f(x) \cdot \sin x \, dx
17 \boldsymbol{\int_a^b f(x) \cdot \sin x \, dx}
18 \quad \pmb{\int_a^b f(x) \cdot \sin x \, dx}
19 \] \[
20 \int_a^b f(x) \cdot \sin x \, dx
21 \boldsymbol{\int_a^b f(x) \cdot \sin x \, dx}
22 \quad \pmb{\int_a^b f(x) \cdot \sin x \, dx}
23 \] \[
24 \int_a^b f(x) \cdot \sin x \, dx
25 \boldsymbol{\int_a^b f(x) \cdot \sin x \, dx}
26 \quad \pmb{\int_a^b f(x) \cdot \sin x \, dx}
27 \]

```

$$\pm aB\beta\Gamma \quad \pm aB\beta\Xi \quad \pm aB\beta\Xi$$

$$\sum_{x \in X} \ln x \quad \sum_{x \in X} \ln x$$

$$\sum_{x \in X} \ln x \quad \sum_{x \in X} \ln x$$

$$\int_a^b f(x) \cdot \sin x \, dx$$

$$\int_a^b f(x) \cdot \sin x \, dx$$

$$\int_a^b f(x) \cdot \sin x \, dx$$

[Buch1/AmSPakete/amsbsyxml.tex]

12.2.3 Paket amsopn

Zusätzliche Operatorennamen und Varianten mit kommen mit dem Paket `amsopn` daher. $\text{\LaTeX}$ stellt die schon in Tabelle 6.6 aufgelisteten Funktionsbezeichnungen zur Verfügung. `amsmath` definiert zwei weitere Funktionsbezeichnungen. Zusätzlich werden Varianten der hinzugekommenen und zweier Funktionen aus dem Standard angeboten. Die Liste der vordefinierten Funktionsbezeichnungen wird in der folgenden Tabelle um die angegebenen erweitert.

In der Regel stehen damit alle Funktionen zur Verfügung, die man beim Formelsatz benötigt. Sollten jedoch zusätzliche Funktionen benötigt werden wie etwa die alternative Funktionsbezeichnung `\tg` statt `\tan`, stellt das Paket `amsopn` Befehle zur Verfügung, um eigene Funktionen und Operatoren zu verwenden.

Tabelle 12.2: Zusätzliche vordefinierte Funktionsbezeichnungen (Operatoren) bei Verwendung von `amsmath`

$\lim$	<code>\varliminf</code>	$\overline{\lim}$	<code>\varlimsup</code>
$\injlim$	<code>\injlim</code>	$\varinjlim$	<code>\varinjlim</code>
$\projlim$	<code>\projlim</code>	$\varprojlim$	<code>\varprojlim</code>

Zwei Varianten des Befehls `\operatorname` stehen zur Verfügung, um eine Teilformel *ausnahmsweise* als Operator zu verwenden. Bei häufigem Gebrauch empfiehlt sich die Definition einer eigenen Funktion oder eines eigenen Operators mit den beiden Varianten des `\DeclareMathOperator`-Befehls.

```
\operatorname*{<Teilformel>}
```

Die Variante `\operatorname` setzt das Argument im mathematischen Modus unter Verwendung des Befehls `\mathrm`. Damit werden Funktionsbezeichnungen in der gleichen Form wie die vordefinierten Funktionen gesetzt. Im Unterschied zum echten mathematischen Modus werden dabei auch die beiden Zeichen `-` und `*` wie im Textmodus gesetzt, nämlich – als Trennzeichen und `*` hochgesetzt statt als Operator.

Die Variante `\operatorname*` unterscheidet sich dadurch, dass angehängte Exponenten und Indizes so gesetzt werden, als wäre ein `\limits`-Befehl eingefügt.

Wird eine Funktion oder ein Operator häufiger verwendet, sollte dafür ein Befehl definiert werden. So erspart man sich auch die häufige Wiederholung des `\operatorname`-Befehls und kann statt dessen einen kürzeren eigenen Befehlsnamen verwenden. Dazu stellt das Paket die beiden Varianten

```
\DeclareMathOperator*{<Befehl>}{<Teilformel>}
```

zur Verfügung. Der Unterschied zwischen `\DeclareMathOperator` und `\DeclareMathOperator*` ist wieder, dass bei der `*`-Form Exponenten und Indizes des Operators oder der Funktion in der `\limits`-Form gesetzt werden. `<Teilformel>` hat die gleiche Bedeutung wie bei `\operatorname`. `<Befehl>` ist ein *neu* zu definierender Befehl, der für die Funktion oder den Operator verwendet wird. Beide Befehle können nur in der Präambel benutzt werden.

Damit ließen sich *in der Präambel* beispielsweise die beiden folgenden Funktionen definieren:

```
\DeclareMathOperator{\tg}{tg}
\DeclareMathOperator*{\Lim}{Lim}
```

Bei `\Lim` wird im Displaymodus ein Index *unter* der Bezeichnung „Lim“ gesetzt und ein Exponent darüber.

Werden die Funktionen nur ausnahmsweise verwendet, können statt einer Definition in der Präambel die Funktionen innerhalb einer Formel als solche gekennzeichnet werden:

```
[\operatorname{tg} x = \tan x\]
[\operatorname*{\Lim}_{n \rightarrow \infty}
f_n(x) = f(x)\]
```

Die Ausgabe unterscheidet sich nicht von der ersten Version mit den in der Präambel vereinbarten Funktionen. In den Formeln schreibt man dann `\tg` statt `\operatorname{tg}` und `\Lim` statt `\operatorname*{\Lim}`.

12.2.4 Texte in Formeln mit amstext

Kurze Texteinschübe ohne Zeilenumbruch können mit dem Paket `amstext` in Formeln mit Hilfe des Befehls `\text` gesetzt werden. Dieses Paket wird von `amsmath` automatisch geladen, kann aber auch ohne das Paket `amsmath` verwendet werden. Der Befehl hat die Form

```
\text{<Text>}
```

und setzt sein Argument in der aktuellen Schriftart für normalen Text. Er kann gegebenenfalls auch im Textmodus verwendet werden. Der Befehl ignoriert allerdings beispielsweise Umlaute und ß, wenn sie mit Hilfe des Pakets `inputenc` als Zeichen eingegeben wurden. In den genannten Fällen ist eine Eingabe in Form der im Paket `babel` zur Verfügung gestellten Form „a, „s oder der entsprechenden Grundformen `\a`, `\ss` möglich.

Der Befehl wurde bereits in mehreren Beispielen verwendet. Daher soll hier ein kurzes Beispiel für den Befehl genügen. Um zu zeigen, dass Text korrekt gesetzt wird, wurde ein aus mehreren Worten bestehender Text in die Formel eingefügt.

```
9 \[
10 f(x)=\ln x \quad
11 \text{f"ur den Fall }
12 x>0
13 \]
```

$$f(x) = \ln x \quad \text{für den Fall } x > 0$$

[Buch1/AmSPakete/text.tex]

12.2.5 Ergänzungen mit `amsxtra`

Das Paket `amsxtra` stellt einige Ergänzungen zur Verfügung, die auf $\mathcal{AMSTeX}$ zurückgehen. Eine erste Gruppe von Befehlen setzt Akzente als Zeichen wie Exponenten, da in mathematischen Formeln gelegentlich auch Akzentzeichen in der Position von Exponenten verwendet werden. Die folgenden Befehle setzen – nach einer Teilformel verwendet – den im Befehlsnamen genannten Akzent in der Position eines Exponenten:

```
\sphot
\spcheck
\sptilde
\spdot
\spddot
\spdddot
\spbreve
```

Der Befehl ergibt sich aus dem Namen des Akzents, dem die Buchstaben `sp` vorangestellt wurden. Zu beachten ist, dass die Befehle das Hochstellen selbst übernehmen und daher kein Zeichen `^` anzugeben ist.

Als weiterer $\mathcal{AMSTeX}$ -Befehl wird

```
\accentedsymbol{<Befehl>}{<Symbol mit Akzent(en)>}
```

vereinbart, der sich von einem statt dessen verwendeten `\newcommand`-Befehl dadurch unterscheidet, dass er das fertig formatierte zusammengesetzte Zeichen statt der Befehle zu Aufbau des Zeichens abspeichert. Dieses Vorgehen sparte kostbare CPU-Zeit, als die Rechner im Vergleich zu heute extrem langsam arbeiteten. Mit modernen Rechnern ist der Mehraufwand durch den wiederholten Aufbau des zusammengesetzten Zeichens vernachlässigbar.

Ist auch das mittlerweile überholte Paket `amstex` geladen, wird zur Kompatibilität mit alten Versionen auch ein Befehl `\fracwithdelims` definiert, der ein Vorgänger des aktuellen Befehls `\genfrac` ist. Da der alte Befehl nicht mehr verwendet werden sollte, wird hier auf eine Beschreibung verzichtet.

Layout-Erweiterungen der $\mathcal{A}\mathcal{M}\mathcal{S}$

Neben den Paketen für den erweiterten Mathematiksatz hat die $\mathcal{A}\mathcal{M}\mathcal{S}$ – die American Mathematical Society – hat die $\mathcal{A}\mathcal{M}\mathcal{S}$ Dokumentklassen für die von ihr herausgegebenen Publikationen schreiben lassen, die uneingeschränkt genutzt werden können.

Die Pakete der $\mathcal{A}\mathcal{M}\mathcal{S}$, mit deren Hilfe die von der $\mathcal{A}\mathcal{M}\mathcal{S}$ entwickelten Fonts genutzt werden können:

Pakete: amsfonts, amssymb, cmmib57, eucal, eufrak, euscript

Auch die folgenden Klassen sind Teil der $\mathcal{A}\mathcal{M}\mathcal{S}$ -Erweiterungen:

Klassen: amsart, amsbook, amsdtx, amsldoc, amsproc

Diese Pakete und ihre Einsatzgebiete werden im Anschluss an eine Kurzbeschreibung in den nachfolgenden Abschnitten ausführlich beschrieben.

Pakete mit stilistischen Alternativen und Ergänzungen

amsthm

Ein sehr interessantes Paket, das dem Anwender gute Kontrolle über Theorem-Umgebungen gibt. Mehrere Theoremstile sind vordefiniert, und eigene können mit diesem Paket vereinbart werden.

amsmidx

Dieses Paket enthält die $\mathcal{A}\mathcal{M}\mathcal{S}$ -Versionen der Befehle zum Erstellen und Setzen eines Stichwortverzeichnisses.

amsrefs

Dieses Paket stellt eine $\text{\LaTeX}$ -basierte Alternative zum Aufbau von Literaturverzeichnissen zur Verfügung. Andere Programme wie $\text{BIB}\text{\TeX}$ werden nicht benötigt. Die Hilfspakete *ifoption*, *pcatcode*, *rkeyval*, *textcmds* und *optional mathscinet* werden automatisch angezogen.

textcmds

Dieses Paket definiert ‚Kurzbefehle‘ für Textsymbole, die üblicherweise in $\text{\LaTeX}$ -Dokumenten verwendet werden. Die Verwendung der Kurzbefehle erleichtert auch die Umwandlung der $\text{\LaTeX}$ -Dokumente in andere Formen wie HTML und macht sie weniger fehleranfällig.

mathscinet

Das Paket definiert eine Reihe von nicht im $\text{\LaTeX}$ -Standard vorgesehenen Befehlen, die gelegentlich in Literaturdaten verwendet werden, die vom MathSciNet exportiert wurden. Da das Paket nur für diese speziellen Literaturdaten gedacht ist, wird es hier nicht behandelt.

Interne Pakete

Interne Pakete sind uninteressant für Anwender und sollten von ihnen nicht geladen werden. Auf diese Pakete wird daher nicht weiter eingegangen.

ifoption

Ein Hilfspaket zur Behandlung von Optionen von Paketen von Michael J. Downes, das beispielsweise von *amsrefs* verwendet wird. Es ist von Interesse für Entwickler von Paketen, enthält aber keine für Anwender interessanten Befehle.

pcatcode

Ein weiteres Hilfspaket von Michael J. Downes, das für Entwickler von Paketen interessant ist, wenn *Catcodes* modifiziert werden, für die ein *Stack* zur Verfügung gestellt wird. Es wird beispielsweise von *amsmath* verwendet.

rkeyval

Bei diesem Hilfspaket von Michael J. Downes und David M. Jones handelt es sich um eine Reimplementierung des Grundpakets *keyval*. Im Unterschied zu *keyval* wird eine restriktivere Syntax verwendet, um eine bessere Fehlerbehandlung zu ermöglichen.

upref

Ein reines Hilfspaket, welches ausschließlich interne Befehle enthält, die ausgelagert wurden, um sie besser warten zu können.

Pakete für die von der $\mathcal{AMS}$ entwickelten Fonts

Namen von Schriften (engl. fonts) werden in diesem Kapitel in Anführungszeichen gesetzt, um dem mit den Namen nicht vertrauten Leser die Zuordnung zu erleichtern.

amsfonts

Dieses Paket ist die zentrale Schnittstelle (engl. interface) zu den von der $\mathcal{AMS}$ zur Verfügung gestellten Fonts. Daneben erlaubt es auch den Zugriff auf einige spezielle Symbole sowie die ‚Blackboard bold‘- und Fraktur-Buchstaben.

amssymb

amssymb fasst die von der $\mathcal{AMS}$ definierten Sonderzeichen zusammen und kann auch eigenständig verwendet werden, wenn die Sonderzeichen in einem Dokument benötigt werden.

cmmib57

Durch Laden dieses Pakets ist es möglich, die PostScript-Versionen der ‚cmmib‘- und ‚cmb57‘-Fonts von Inc. Y&Y und Inc. Blue Sky Research statt der Standard-Rasterfonts zu verwenden.

eufrak

Erlaubt die Verwendung der Fraktur-Buchstaben im mathematischen Modus, wenn das Paket amsfonts nicht geladen wird. Die Schrift Fraktur wurde von Herrmann Zapf, einem international bekannten Schriftdesigner aus Deutschland, im Auftrag der $\mathcal{AMS}$ entwickelt.

eucal

Ersetzt den beim Befehl `\mathcal` verwendeten kalligrafischen Font durch die Schrift ‚Euler Script‘. Dieser kalligrafische Font wurden von Herrmann Zapf im Auftrag der $\mathcal{AMS}$ entworfen. Die Zeichen haben eine handschriftliche Note und sind daher sehr beliebt.

euscript

euscript ist ein zu eucal identisches Paket, das früher (nur) unter diesem Namen verfügbar war und aus Gründen der Kompatibilität auch weiterhin angeboten wird.

13.1 Dokumentklassen und stilistische Ergänzungen

Für ihre Veröffentlichungen hat die $\mathcal{AMS}$ eine Reihe von Klassen und Paketen entwickelt, die auch der Allgemeinheit zur Verfügung stehen. Die Klassen für die verschiedenen Veröffentlichungsreihen basieren auf drei Grundklassen, die Autoren beim Erstellen verwenden sollten.

Die Pakete wurden zur Unterstützung dieser Klassen entwickelt. Eine Reihe von ihnen ist nur im Zusammenhang mit den Klassen oder für Entwickler von Klassen und Paketen von Bedeutung. Interessant für Anwender sind speziell die Pakete `amsthm`, `textcmds` und `amsrefs`, die auch im Folgenden behandelt werden. Auch vorgestellt wird das Paket `amsidx` für die Behandlung mehrerer Register, das bisher nur im Zusammenhang mit der Klasse `amsbook` von Interesse ist. Dieses sehr neue Paket könnte im Rahmen der weiteren Entwicklung auch für andere Anwender interessant werden, wenn die Abhängigkeiten von `amsbook` beseitigt werden (was leicht möglich ist). Daher wird auch dieses Paket vorgestellt.

13.1.1 Die Dokumentklassen der $\mathcal{AMS}$

Die $\mathcal{AMS}$ veröffentlicht in einem eigenen Verlag Bücher und Zeitschriften. Für diese hat die $\mathcal{AMS}$ Dokumentklassen entwickelt, die für verschiedene Buchreihen und Zeitschriften angepasst sind. Diese Klassen basieren alle auf den drei Grundklassen `amsart`, `amsbook` und `amsproc`, die im Folgenden kurz vorgestellt werden. Hinzu kommen zwei weitere, hier nicht behandelte Klassen `amstdx` für die Dokumentation von Klassen und Paketen und `amsldoc` für Benutzeranleitungen.

Die Grundklassen sind für die folgenden Veröffentlichungen gedacht:

- amsart** für Zeitschriftenartikel,
- amsbook** für Bücher (Monografien),
- amsproc** für Tagungsbände, Proceedings.

Alle drei Klassen laden nach Voreinstellung automatisch die Pakete `amsmath`, das seinerseits `amstext`, `amsbsy` und `amsopn` anzieht, und `amsfonts`. Auch das Hilfspaket `amsgen` wird verwendet. Das Paket `amsthm` ist sogar in die Klassen integriert.

Die vorgestellten Klassen haben viele Optionen mit den Standardklassen `article`, `report` und `book` gemeinsam. Allerdings werden nicht alle Optionen

der Standardklassen unterstützt. Auf der anderen Seite kommen neue Optionen hinzu.

Die folgende Tabelle listet alle Optionen zusammengefasst nach Wirkungsbereichen auf:

Tabelle 13.1: Optionen der $\mathcal{A}\mathcal{M}\mathcal{S}$ -Klassen

Papiergrößen und Ausrichtung	a4paper, letterpaper, landscape, portrait
Ein- oder zweiseitiger Druck	oneside, twoside
Vorläufige oder endgültige Version	draft, final
Elektronische Version	e-only
Ein- oder zweispaltiger Satz	onecolumn, twocolumn
Nummerierung von Formeln	leqno, fleqn, reqno, centertags, tbtags
Grundschriftgrößen	8pt, 9pt, 10pt, 11pt, 12pt
Schriftarten	noamsfonts, psamsfonts
Titelseite	titlepage, notitlepage
Startseite von Kapiteln	openright, openany (nur amsbook)
Mathematische Erweiterung	nomath
Index (redundante Option)	makeidx

Die Standardoptionen sind beschrieben in 3.1.1 (S. 49), die hinzugekommenen Optionen haben im Einzelnen folgende Bedeutung:

Ausrichtung

portrait

Bei den Standardklassen fehlendes Gegenteil zu `landscape`.

Elektronische Veröffentlichungen (**nur amsart**)

e-only

Option für ausschließlich elektronisch veröffentlichte Artikel, bei denen einige bibliografische Angaben von denen (auch) gedruckter Artikel abweichen.

Mathematische Erweiterungen

nomath

Unterdrückt das Laden von `amsmath`.

Fontoptionen

noamsfonts

Verhindert das Laden des Pakets `amsfonts`.

psamsfonts

Die Option wird weitergegeben an das Paket `amsfonts`, welches dann die PostScript-Version der $\mathcal{AMS}$ -Fonts statt der Rasterfonts lädt.

Nummerierung von Formeln

reqno

Gleichungsnummern werden rechts gesetzt (voreingestellt ist bei den $\mathcal{AMS}$ -Klassen links).

centertags

Ausgabe der Gleichungsnummern bei mehrzeiligen Formeln vertikal zentriert.

tbtags

Ausgabe der Gleichungsnummern bei mehrzeiligen Formeln in der ersten Zeile bei Formelnummern links und in der letzten Zeile bei Formelnummern rechts.

Schriftgrößen

8pt Legt die Schriftgröße 8pt als Grundgröße fest (`\normalsize`).

9pt Legt die Schriftgröße 9pt als Grundgröße fest (`\normalsize`).

Redundant

makeidx

Ohne Wirkung, da die Klassen ohnehin die Funktionalität des Pakets `makeidx` beinhalten.

Voreinstellungen

Die folgenden Option entsprechen der Voreinstellung in allen $\mathcal{AMS}$ -Dokumentklassen: `leqno`, `centertags`, `letterpaper`, `portrait`, `10pt`, `twoside`, `onecolumn`, `final`.

Die Pakete definieren eine Reihe zusätzlicher Befehle, die zu einem großen Teil zusätzliche Informationen für den Titel und die Adressen von Autoren festlegen. Da diese Zusatzinformationen nur für Veröffentlichungen bei der $\mathcal{AMS}$ von Bedeutung sind und auch fehlen können, werden sie hier nicht vorgestellt.

Zwei zusätzlich definierte Befehle zum Umschalten von Schriftgrößen dürften jedoch von allgemeinerem Interesse sein, zumal die Klassen auch Schriftgrößen unterstützen, die die Standardklassen ignorieren. In den $\mathcal{AMS}$ -Klassen sind die folgenden *elf* Schriftgrößen definiert:

<code>\normalsize</code>	<code>\small</code>	<code>\Small</code>	<code>\SMALL</code>
<code>\tiny</code>	<code>\Tiny</code>	<code>\large</code>	<code>\Large</code>
<code>\LARGE</code>	<code>\huge</code>	<code>\Huge</code>	

Die in aufsteigender Reihenfolge angegebenen Schriftgrößen können sowohl mit diesen Befehlen explizit ausgewählt, als auch relativ verkleinert oder vergrößert werden:

```
\larger[<Stufen>]
\smaller[<Stufen>]
```

Bei fehlender Angabe wird die Schrift um eine Stufe verändert, also zur nächstgrößeren (`\larger`) oder nächstkleineren (`\smaller`) Schrift gewechselt (Stufe = 1). Bei Vergrößerungen oder Verkleinerungen um mehr als eine Stufe ist die Anzahl der Stufen als optionales Argument anzugeben.

13.1.2 Paket amsthm

Dieses Paket der American Mathematical Society erweitert den `\newtheorem`-Befehl von $\text{\LaTeX}$. Wie im Paket `theorem` von Frank Mittelbach (siehe 9.19 auf Seite 294) berücksichtigt `\newtheorem` einen aktuell eingestellten Stil. Hinsichtlich der Stile bestehen allerdings erhebliche Unterschiede, so dass ein Austausch der Pakete nicht möglich ist. Ein weiterer Unterschied liegt darin, dass `amsthm` Theorem-Umgebungen ohne Nummerierung unterstützt. Eine Umgebung für Beweise ist vordefiniert. Die Dokumentklassen der $\mathcal{AMS}$ beinhalten dieses Paket, d. h. es muss bei `amsart`, `amsrep` und `amsbook` nicht extra geladen werden.

Beim Laden des Pakets für andere Dokumentklassen mit

```
\usepackage[<Stil-Datei(en)>]{amsthm}
```

ist zu beachten, dass es erst *nach* dem Paket `amsmath` geladen werden darf, falls beide Pakete geladen werden. Optional können Namen von einer oder mehreren Stil-Dateien angegeben werden, die um die Dateinamenserweiterung `.thm` ergänzt und eingelesen werden. Es wird angenommen, dass diese Datei(en) eigene Theoremstile definieren, wie es im Folgenden beschrieben ist.

13.1.2.1 Vereinbarung von Theorem-Umgebungen

Vom Paket wird lediglich eine Umgebung `proof` für Beweise vordefiniert. Umgebungen für Theoreme, Sätze, Lemmas, ... muss der Anwender wie bei Standard- $\text{\LaTeX}$ mit dem bereits bekannten `\newtheorem`-Befehl vereinbaren, bevor sie verwendet werden können. Im Unterschied zu Standard- $\text{\LaTeX}$ berücksichtigt `\newtheorem` allerdings den voreingestellten oder zuletzt festgelegten Theoremstil.

Zur Vereinbarung von Theorem-Umgebungen ohne Nummerierung steht der Befehl

```
\newtheorem*{<UName>}{<Bezeichnung>}
```

zur Verfügung. Die beiden Varianten des optionalen Arguments für die Nummerierung entfallen hier, da die so erzeugten Theorem-Umgebungen nicht nummeriert werden. Der aktuelle Theoremstil wird wie beim `\newtheorem`-Befehl berücksichtigt.

13.1.2.2 Anwenden von Theorem-Umgebungen

Unter Verwendung dieses Pakets vereinbarte Theorem-Umgebungen erlauben im Unterschied zu Standard- $\text{\LaTeX}$ die Angabe eines optionalen Arguments zur Vereinbarung zusätzlicher Informationen, beispielsweise eines Literaturverweises. Das optionale Argument wird in der Regel nach Nummer und Bezeichnung in runde Klammern gesetzt (lässt sich bei eigenen Stilen ändern). Nach dieser Überschrift kommt ein abschließendes Satzzeichen (bei eigenen Theoremstilen, falls eingestellt, auch ein Zeilenumbruch). Mit einer zuvor vereinbarten Theorem-Umgebung `<UName>` kann sie wie folgt verwendet werden:

```
\begin{<UName>}[<Zusatzinfo>]
  <Text> ...
\end{<UName>}
```

Vordefiniert ist die Umgebung `proof` zum Setzen von Beweisen. Sie ist nicht nummeriert und hat die folgende Form:

```
\begin{proof}[<Überschrift>]
...
\end{proof}
```

Abweichend von den sonstigen Voreinstellungen *ersetzt* das optionale Argument beim Öffnen einer Umgebung `proof` die voreingestellte Überschrift *Proof*. Die Umgebung ist für nicht zu lange Beweise gedacht (nicht länger als ein bis zwei Seiten). Längere Beweise werden besser als `\section` oder `\subsection` gesetzt. Am Ende der Umgebung wird automatisch ein QED-Symbol gesetzt, das üblicherweise das Ende des Beweises markiert.

Die Überschrift und das QED-Symbol lassen sich leicht mit einem `\renewcommand`-Befehl ändern, da sie durch die beiden folgenden Befehle festgelegt werden:

```
\proofname
```

Überschrift des Beweises.
(Voreinstellung: „Proof“)

```
\qedsymbol
```

QED-Symbol zum Markieren des Beweisendes.
(Voreinstellung: ein kleines Quadrat)

Das QED-Symbol entspricht auch den deutschen Konventionen, so dass lediglich die Überschrift mit

```
\renewcommand{\proofname}{Beweis}
```

geändert werden muss.

Steht am Ende des Beweises eine Formel, eine Liste oder eine vergleichbare Umgebung, ist die Positionierung des QED-Symbols häufig problematisch. Aus diesem Grund ist der Befehl

```
\qedhere
```

vorgesehen, mit dem der Autor das QED-Symbol explizit positionieren kann. Bei Verwendung einer aktuellen Version des Pakets `amsmath` (das Paket `amsthm` darf erst *danach* geladen werden!) wird das QED-Symbol rechtsbündig gesetzt. Sollten bei Angabe in einer Formel Fehler auftreten (etwa bei älteren Versionen von `amsmath`), lässt sich das Problem meist durch `\mbox{\qedhere}` statt `\qedhere` lösen.

Zum Abschluss langer Beweise, die als `\section` oder `\subsection` gesetzt werden, ist der Befehl

```
\qed
```

vorgesehen. Der Befehl setzt das QED-Symbol außerhalb einer `proof`-Umgebung.

13.1.2.3 Theoremstile

Vorhandene Theoremstile werden mit

```
\theoremstyle{<Stil>}
```

aktiviert und von nachfolgenden `\newtheorem`-Befehlen berücksichtigt. Ohne einen `\theoremstyle`-Befehl wird der voreingestellte Stil `plain` verwendet.

Vordefiniert sind drei Theoremstile, die unterschiedlich stark hervorgehoben sind. Die vordefinierten Stile sind:

plain

für Theoreme, Sätze, Lemmas, Korollare, ..., die deutlich vom übrigen Text abgesetzt werden sollen;
(Voreinstellung bei fehlendem `\theoremstyle`-Befehl)

definition

für Definitionen, Bedingungen, Aufgaben, Beispiele ..., die weniger deutlich vom übrigen Text abgesetzt sein sollen;

remark

für Bemerkungen, Anmerkungen, Zusammenfassungen, ..., die sich noch weniger vom anderen Text unterscheiden sollen.

Die Aktivierung eines Stils für nachfolgende `\newtheorem`-Befehle erfolgt durch Angabe des Stils als Argument eines `\theoremstyle`-Befehls.

Entsprechend der Voreinstellung wird die Nummer eines ‚Theorems‘ *nach* der bei der Vereinbarung festgelegten Bezeichnung gesetzt. Es ist aber auch nicht unüblich, die Reihenfolge zu vertauschen und die Nummer *vor* die Bezeichnung zu setzen. Diese Variante wird durch den Befehl

```
\swapnumbers
```

unterstützt, der für danach vereinbarte Theorem-Umgebungen die Reihenfolge vertauscht (der Befehl wirkt als Schalter).

Entsprechen die vordefinierten Theoremstile nicht den eigenen Vorstellungen, kann man mit dem Befehl

```
\newtheoremstyle{<Name>}{<Abst_v>}{<Abst_h>}{<Font>}%
  {<Einzug>}{<ÜFont>}{<Trenn>}{<Abst_Ü>}{<Ü_Bef>}
```

eigene Theoremstile vereinbaren. Die neun anzugebenden Argumente haben dabei die folgende Bedeutung:

- <Name>** Name bestehend aus Buchstaben für den vereinbarten Stil;
- <Abst_v>** Abstand eines ‚Theorems‘ in diesem Stil vom vorhergehenden Text;
(Längenangabe)
- <Abst_h>** Abstand eines ‚Theorems‘ in diesem Stil zum nachfolgenden Text;
(Längenangabe)
- ** Einstellung des Fonts, in dem der Text des ‚Theorems‘ gesetzt wird, beispielsweise `\itshape`;
(Voreinstellung: `\itshape`)
- <Einzug>** Einrückung der Überschrift eines ‚Theorems‘ in diesem Stil;
(Längenangabe oder leer, falls nicht eingerückt werden soll)
- <ÜFont>** Einstellung des Fonts für die Überschrift;
(Voreinstellung: `\bfseries`)
- <Trenn>** ‚Trennzeichen‘ nach der Überschrift, beispielsweise ein Doppelpunkt;
(Voreinstellung: `.`)
- <Abst_Ü>** Abstand zwischen der Überschrift und dem Text des ‚Theorems‘;
(Längenangabe, Leerzeichen (normaler Wortabstand) oder `\newline` für einen Zeilenumbruch)
- <Ü_Bef>** Befehle zum Setzen der Überschrift über das Theorem; unter Verwendung eines Makros mit drei Argumenten werden die Befehle ausgeführt:
 - #1** durch `\newtheorem` festgelegte Bezeichnung;
 - #2** aktuelle Nummer des ‚Theorems‘;
 - #3** ein beim Öffnen der Umgebung angegebenes optionales Argument.

Ist dieses Argument leer (das ist eigentlich der Normalfall),
wird die Überschrift gemäß den Voreinstellungen gesetzt.

```

9  \newtheoremstyle{mdef}{3ex}{3ex}{\slshape}{}%
10 % UeFont Trenn AbstUe    UeBefs
11 {}      {:}      {\newline}{}
12 \theoremstyle{mdef}
13 \newtheorem{mthm}{Guter Tipp}[section]
14 \newtheorem*{mthms}{Noch besserer Tipp}
15 \begin{document}
16 \thispagestyle{empty} % für Buch
17 \setcounter{section}{5} % Nummer erhöhen
18 Etwas Text vor dem selbstdefinierten Theorem, damit man
19 den vereinbarten Abstand von 3\,ex sieht.
20 \begin{mthm}[Aus der Hexenk"uche]
21   Ein Text im neu definierten Theorem \texttt{mthm}.
22   \(\ 1+1=2 \)
23 \end{mthm}
24 \begin{mthm}[Aus der Hexenk"uche]
25   Die Wissenschaft hat festgestellt
26   \(\ 1+1 \) ist doch \(\ {}=3 \).
27 \end{mthm}
28 Etwas Text nach dem selbstdefinierten Theorem.
29 \begin{mthms}
30   Theorem-Umgebung ohne Nummer mit Namen \texttt{mthms}.
31 \end{mthms}

```

Etwas Text vor dem selbstdefinierten Theorem, damit man den vereinbarten Abstand von 3 ex sieht.

Guter Tipp 5.1 (Aus der Hexenküche):

Ein Text im neu definierten Theorem $mthm$. $1 + 1 = 2$

Guter Tipp 5.2 (Aus der Hexenküche):

Die Wissenschaft hat festgestellt $1 + 1$ ist doch $= 3$.

Etwas Text nach dem selbstdefinierten Theorem.

Noch besserer Tipp:

Theorem-Umgebung ohne Nummer mit Namen $mthms$.

13.1.3 Paket textcmds

Dieses Paket von Michael J. Downes definiert Abkürzungen für häufiger verwendete $\text{\LaTeX}$ -Befehle zum Setzen von Sonderzeichen. Es wird geladen mit

```
\usepackage{textcmds}
```

und stellt die in der folgenden Tabelle aufgeführten Kurzbefehle zur Verfügung. Die Tabelle gibt zu jeder Abkürzung in der Spalte ‚ $\text{\LaTeX}$ -Befehl‘ den Original-Befehl und in Spalte ‚Zeichen‘ das gesetzte Zeichen an.

	$\text{\LaTeX}$-Befehl	Zeichen
<code>\mdash</code>	<code>\textemdash</code>	—
<code>\ndash</code>	<code>\textendash</code>	–
<code>\qd</code>	<code>\textquestiondown</code>	¿
<code>\xd</code>	<code>\textexclamdown</code>	¡
<code>\ldq</code>	<code>\textquotedblleft</code>	“
<code>\rdq</code>	<code>\textquotedblright</code>	”
<code>\lsq</code>	<code>\textquoteleft</code>	‘
<code>\rsq</code>	<code>\textquoteright</code>	’
<code>\bul</code>	<code>\textbullet</code>	•
<code>\vsp</code>	<code>\textvisiblespace</code>	␣
<code>\pdc</code>	<code>\textperiodcentered</code>	·
<code>\vrt</code>	<code>\textbar</code>	
<code>\cir</code>	<code>\textasciicircum</code>	^
<code>\til</code>	<code>\textasciitilde</code>	~
<code>\bsl</code>	<code>\textbackslash</code>	\
<code>\cwm</code>	<code>\textcompwordmark</code>	◌

Zusätzlich zu den Abkürzungen definiert das Paket einige weitere nützliche Befehle. Um Texte in spitze Klammern oder doppelten Anführungszeichen zu setzen, gibt es die beiden Befehle

```
\lara{<Text>}
\qq{<Text>}
```

`\lara` schließt $\langle \textit{Text} \rangle$ in spitze Klammern, `\qq` in doppelte Anführungszeichen ein.

Zwei weitere Befehle dienen zum Hoch- und Tiefstellen von Texten im Textmodus:

```
\tsup{<Hochgestellter Text>}
\tsub{<Tiefgestellter Text>}
```

`\tsup` entspricht $\hat{\dots}$ im mathematischen Modus und `\tsub` entspricht $_{\dots}$, nur dass die Befehle ebenso wie ihre Argument im Textmodus gesetzt werden.

Schließlich wird das im mathematischen Modus vorhandene Zeichen `\prime` ($\prime$) unter der Bezeichnung

```
\tprime
```

für den Textmodus verfügbar gemacht.

Die folgende Tabelle zeigt jeweils ein Beispiel für die fünf zusätzlichen Befehle.

Eingabe	Ergebnis
<code>\lara{Text in spitzen Klammern}</code>	$\langle \text{Text in spitzen Klammern} \rangle$
<code>\qq{Text in Anföijhrungszeichen}</code>	“Text in Anföijhrungszeichen”
Dies ist <code>\tsup{hochgestellter Text}</code>	Dies ist ^{hochgestellter Text}
Dies ist <code>\tsub{tiefgestellter Text}</code>	Dies ist _{tiefgestellter Text}
<code>a\tprime b</code>	$a'b$

13.1.4 Mehrere Register mit `amsmidx`

Als Ergänzung speziell zur $\mathcal{A}\mathcal{M}\mathcal{S}$ -Klasse `amsbook` hat Barbara Beeton aus dem Paket `multind` das neue Paket `amsmidx` entwickelt, das mehrere Register in einem Dokument ermöglicht. Die Nutzung mit anderen Klassen ist (bisher) dadurch eingeschränkt, dass ein Befehl redefiniert wird, der nur in dieser speziellen Dokumentklasse definiert ist. Selbst in den Klassen `amsart` und `amsbook` fehlt dieser spezielle Befehl, der zum Setzen mehrerer Register allerdings auch nicht notwendig ist. Durch Definition aller fehlenden Befehle vor dem Laden des Pakets mit

```
\usepackage{amsmidx}
```

lässt sich das Paket gegebenenfalls auch mit anderen Klassen nutzen.

Das Paket hat Ähnlichkeiten mit dem Paket `index` von David M. Jones, das aus dem genannten Grund für andere Dokumentklassen zum jetzigen Zeitpunkt besser geeignet ist. Aufgrund einer abweichenden Organisation – bei `amsidx` unterscheiden sich die Dateinamen für die einzelnen Register bei gleichen Erweiterungen der Dateinamen, bei `index` unterscheiden sich nur die Erweiterungen des Dateinamens – ist der organisatorische Aufwand etwas geringer als bei `index` bei sehr ähnlichen Befehlen für die Einträge in die Register. Die Aufbereitung der Register für den Satz ist – abgesehen von den unterschiedlichen Dateinamen – gleich.

Jedes zu erstellende Register muss nach dem Laden des Pakets noch *in der Präambel* (vor `\begin{document}`) mit einem Befehl

```
\makeindex{<Indexdatei>}
```

initialisiert werden. `<Indexdatei>` ist dabei ein eindeutiger Name für dieses spezielle Register. Indexeinträge für dieses Register werden in die Datei `<Indexdatei>.idx` geschrieben. Bei mehreren Registern müssen also unterschiedliche Namen angegeben werden.

Einträge in ein Register werden mit dem Befehl

```
\index{<Indexdatei>}{Eintrag}
```

vorgenommen. `<Indexdatei>` ist der Name einer in der Präambel initialisierten Indexdatei. Das zweite Argument ist der Indexeintrag, der die gleiche Form wie das Argument beim `\index`-Befehl des Pakets `makeidx` hat.

Beim Formatieren werden die Einträge in die Indexdatei(en) geschrieben, und zwar in der Reihenfolge, in der sie anfallen. Die Aufbereitung einer Datei `<Indexdatei>.idx` zu einem satzfertigen Register `<Indexdatei>.ind` kann beispielsweise mit `MakeIndex` erfolgen.

An der Stelle, an der das Register `<Indexdatei>` gesetzt werden soll, ist im Dokument der Befehl

```
\Printindex{<Indexdatei>}{<Überschrift>}
```

anzugeben. `<Überschrift>` wird als Überschrift über das Register geschrieben. Um ein Register mit der Standardüberschrift „Index“ zu setzen, kann der verkürzte Befehl

```
\printindex{<Indexdatei>}
```

verwendet werden.

13.2 Literaturverzeichnisse mit `amsrefs`

Bis vor wenigen Jahren musste man das Programm `BIBTEX` verwenden, wenn man die Literaturdaten in datenbankähnlichen Dateien ablegen wollte, um nicht in jedem Dokument manuell ein Literaturverzeichnis aufbauen zu müssen. `BIBTEX` sucht aus den angegebenen ‚Datenbanken‘ die zitierten Literatureinträge heraus und baut daraus eine Umgebung `thebibliography` auf, die dann von `LATEX` gesetzt wird. Zur Auflösung der Literaturverweise muss das Dokument bei dieser Vorgehensweise dreimal formatiert werden.

Das Paket `amsrefs` der $\mathcal{A}\mathcal{M}\mathcal{S}$ von Michael J. Downes bietet nun eine Alternative zu diesem Vorgehen. Es ist damit möglich, Literaturverzeichnisse aus vorliegenden Literaturdaten beim erstmaligen Formatieren mit `LATEX` aufzubauen und die Literaturverweise bereits beim zweiten Formatieren aufzulösen. Externe Programme wie `BIBTEX` werden dabei nicht benötigt. Die Literaturdaten können aus einer speziellen `LATEX`-Datei oder einem vorhandenen `LATEX`-Dokument übernommen werden. Der ‚klassische‘ Weg über `BIBTEX` wird von diesem Paket ebenfalls unterstützt.

Das Paket wird in der üblichen Weise mit dem Befehl

```
\usepackage[Option(en)]{amsrefs}
```

geladen. Dabei können die folgenden Optionen angegeben werden:

- ? zeigt die von `amsrefs` verwendeten Hilfspakete, die verfügbaren Optionen und die Voreinstellungen an.

Zitieren

`alphabetic` | `shortalphabetic` | `author-year` | `y2k` | `numeric`

Durch Angabe einer dieser Optionen kann einer der fünf im Paket vordefinierten Stile für das Literaturverzeichnis gewählt werden (Voreinstellung: `numeric`). Falls keiner der Stile passt und sich auch nicht durch kleinere Änderungen anpassen lässt, kann auch ein eigener Stil definiert werden (vgl. 13.2.8).

`numeric` nummeriert die Literatur und verweist auf die zugehörige Nummer. `alphabetic` verwendet statt einer Nummer einen aus den Anfangsbuchstaben von Autorennamen und Erscheinungsjahr zusammengesetzten Zitiermarke (ähnlich dem BibTeX-Stil `alpha`) zum Verweis auf die Literatur. In der Regel werden nur die beiden letzten Ziffern des Erscheinungsjahres verwendet. `shortalphabetic` entspricht `alphabetic`, verwendet aber kürzere Zitiermarken, bei denen nur der erste Buchstabe eines Autorennamens berücksichtigt wird. `y2k` verhält sich ebenfalls wie `alphabetic`, verwendet aber bei den Zitiermarken die vollständige Jahreszahl. `author-year` verwendete keine Zitiermarken, sondern gibt als Verweis den (ersten) Autor und das Jahr an; dementsprechend enthält auch das Literaturverzeichnis keine Zitiermarken.

Mehrfachzitate

Diese Optionen sind nur im Fall einer nummerierten Literaturliste wirksam.

`compressed-cites` | `non-compressed-cites`

Im Fall sortierter Mehrfachverweise (Option `sorted-cites`) fasst die voreingestellte Option `compressed-cites` drei oder mehr direkt aufeinander folgenden Nummern wie in [1, 9, 10, 11, 12, 16] zu einer Bereichsangabe [1, 9–12, 16] zusammen. Durch Angabe von `non-compressed-cites` lässt sich das Zusammenfassen unterdrücken. Ist das Paket `hyperref` geladen, werden Bereiche nicht zusammengefasst.

`sorted-cites` | `non-sorted-cites`

Im Fall von Mehrfachverweisen mit `\cites` oder `\citelist` bewirkt die voreingestellte Option `sorted-cites` ein aufsteigendes Sortieren der Verweise. `non-sorted-cites` unterdrückt das Sortieren und zugleich das Zusammenfassen von Bereichen (Option `compressed-cites`).

Abkürzungen

`abbrev`

aktiviert alle folgenden Abkürzungs-Optionen.

`initials`

kürzt die Vornamen von Autoren, Editoren und Übersetzern durch ihre Initialen ab. Kompliziertere Fälle müssen als Stil-Varianten im Einzelnen spezifiziert werden (vgl. 13.2.1).

`short-journals`

verwendet Abkürzungen für Namen von Zeitschriften statt der vollständigen Namen. Das ist allerdings nur dann möglich, wenn

mit dem Befehl `\DefineJournal` zuvor Zeitschriftenbezeichnungen festgelegt wurden, die den abgekürzten und vollständigen Namen enthalten.

short-months

kürzt Monatsnamen ab. Voraussetzung ist, dass alle Daten im ISO 8601 Format angegeben sind, also etwa für den 29. Februar 1968 in einer der folgenden Formen:

```
date={1968-02-29}
date={1968-02}
date={1968}
```

Die zweite Gruppe kann neben der Angabe des Monats (Wert 01–12) auch zur Angabe der Jahreszeit verwendet werden. Die Werte 13, 14, 15 und 16 stehen dabei für die Jahreszeiten Winter (engl. *winter*), ‚Spring‘ (Frühjahr), ‚Summer‘ (Sommer) und ‚Fall‘ (Herbst).

In der dieser Beschreibung zu Grunde liegenden Version sind die Monatsnamen und Jahreszeiten im Paket fest in Englisch vorgegeben.

short-publishers

verwendet Abkürzungen für Namen von Verlagen statt der vollständigen Namen. Das ist allerdings nur dann möglich, wenn mit dem Befehl `\DefinePublisher` zuvor Verlagsbezeichnungen vergeben wurden, die den abgekürzten und vollständigen Namen enthalten.

Weitere Optionen

backrefs

erzeugt in Zusammenarbeit mit dem Paket `hyperref` zu den Einträgen des Literaturverzeichnisses eine Liste der Rückverweise auf die Stellen, an denen der Eintrag zitiert wird. Die Liste wird an den Eintrag angehängt.

bibtex-style

Ohne Angabe dieser Option formatiert `amsrefs` die Einträge im Literaturverzeichnis entsprechend den Konventionen der $\mathcal{A}\mathcal{M}\mathcal{S}$ für ihre Veröffentlichungen. Diese Option veranlasst `amsrefs`, die Einträge ähnlich wie bei den `BIBTEX`-Stilen `plain`, `alpha`, `abbrv` oder `unsrt` zu formatieren, je nach angegebenen Zitier- und Abkürzungs-Optionen.

citation-order | sorted

Im Fall der beschriebenen Version 1.23 (vom 28.02.2002) sind diese Optionen nur dann wirksam, wenn das Literaturverzeichnis mit

Hilfe von `BIBTeX` zusammengestellt wird. In diesem Fall wird das Literaturverzeichnis bei Angabe von `sorted` sortiert. `citation-order` ordnet das Literaturverzeichnis in der Reihenfolge des Zitierens.

Bei der Übernahme der Literaturdaten aus einer `LaTeX`-Datei wird die dort vorgefundene Reihenfolge bei der beschriebenen Version beibehalten. Die Sortierung sollte daher bei den Originaldaten vorgenommen werden.

lite unterdrückt das Laden der Hilfspakete `textcmds` (definiert mehrere Kurzbefehle und kann dadurch Konflikte mit anderen Paketen oder selbstdefinierten Befehlen verursachen) und `mathscinet` (definiert mehrere ‚Special Characters‘ und Akzente).

nobysame

Normalerweise werden bei mehreren aufeinanderfolgenden Einträgen im Literaturverzeichnis mit den gleichen Autoren die Autorennamen ab dem zweiten Eintrag durch eine horizontale Linie (____) ersetzt. `nobysame` verhindert das Ersetzen und veranlasst, dass auch in diesen Fällen die Autorennamen gesetzt werden.

traditional-quotes | logical-quotes

legt die Positionierung von Anführungszeichen bei einem nachfolgenden Satzzeichen fest. Beim Einfügen von Anführungszeichen in Felder mit dem Befehl `\bibquotes` (vgl. 478) wird im Fall der voreingestellten traditionellen Variante (`traditional-quotes`) das schließende Anführungszeichen hinter ein direkt folgendes Satzzeichen verschoben. Im Fall von `logical-quotes` folgt das schließende Anführungszeichen hingegen direkt dem gesetzten Inhalt, so dass ein folgendes Satzzeichen außerhalb der Anführungszeichen steht.

Kompatibilität mit früheren Versionen

Die folgenden beiden Option stehen nur zur Verfügung, um vorhandene Dokumente ohne Änderungen formatieren zu können, die alte Versionen von `amsrefs` verwenden. Daher wird eine Warnung ausgegeben, wenn sie angegeben sind.

beta lädt statt der aktuellen Version von `amsrefs` die – nur teilweise kompatible – frühere Version 1.03, um auf dieser Version basierende Dokumente formatieren zu können. Mit neu erstellten Dokumenten soll diese Option nicht (mehr) verwendet werden.

jpa wurde in früheren Versionen angegeben, um das Paket `amsjpa` zu laden, und steht nur noch aus Gründen der Kompatibilität zur Verfügung. Statt dessen sollte der Befehl `\usepackage{amsjpa}` im Dokument zum Laden des Pakets verwendet werden.

13.2.1 Namen

BIB_TE_X erlaubt die Angabe von Eigennamen in unterschiedlichen Formen, etwa als `<Vorname(n) Nachname>` oder `<Nachname>`, `<Vorname(n)>`, `<Zusätze>`. In der ersten dieser beiden Formen ist eine Unterscheidung zwischen Vorname(n) und Nachname nur anhand der Position möglich. Ist wie bei den Chinesen die Reihenfolge vertauscht, wird bei dieser Form der Nachname nicht erkannt, und Bibliografistyle, die Vornamen abkürzen, werden zwangsläufig den Nachnamen statt des Vornamens abkürzen. Um diese Schwierigkeiten zu vermeiden, schreibt `amsrefs` grundsätzlich die Angabe von Eigennamen in der zweiten der beispielhaft genannten Formen vor:

`<Nachname>`, `<Vorname(n)>`, `<Zusätze>`

`<Zusätze>` enthält dabei Angaben wie `Jr.` oder `III`.

Ein weiteres Problem beim Abkürzen besteht darin, dass bei Transkriptionen beispielsweise russischer oder griechischer Namen gelegentlich ein einzelner Buchstabe durch mehrere ersetzt wird oder bei sehr speziellen Namen eine Abkürzung durch den ersten Buchstaben nicht angebracht ist. Beispielsweise stehen beim Namen *Yuri Gagarin* die beiden Buchstaben *Yu* für einen einzelnen kyrillischen Buchstaben, und daher sollte die Kurzform *Yu. Gagarin* und nicht *Y. Gagarin* lauten. Ein Beispiel für einen besonderen Namen, bei dem mehrere Buchstaben eines Vornamens beim Abkürzen berücksichtigt werden sollten, ist *Nicholas deBelleville Katzenbach*, dessen korrekte Kurzform *N. deB. Katzenbach* ist. Das Problem wird dadurch gelöst, dass die beim Abkürzen zu verwendenden Buchstaben in eine geschweifte Klammer gesetzt werden:

```
author{Gagarin, {Yu}ri}
author{Katzenbach, {deB}elleville}
```

Werden Namen mit abgekürzten Vornamen angegeben (abgeschlossen durch einen Punkt), erkennt `amsrefs` am abschließenden Punkt, dass es sich bereits um eine Abkürzung handelt, und kürzt nicht weiter ab. In diesem Fall können die geschweiften Klammern daher entfallen.

Damit sind die Probleme beim Abkürzen von Vornamen gelöst, aber Vor- und Nachname werden beispielsweise bei chinesischen Namen in der falschen Reihenfolge gesetzt. Um das zu vermeiden, bietet `amsrefs` die

Möglichkeit, bei Angabe ein solchen Namens ein spezielles Attribut zu setzen (vgl. 13.2.5).

Abgesehen von der in Ausnahmefällen erforderlichen Gruppierung, können die Namen in der Regel in der normalen $\text{\LaTeX}$ -Schreibweise eingegeben werden. Die Dokumentation Jones (2004) nennt ausdrücklich die folgenden unkritischen Zeichen:

- Die 26 ASCII-Buchstaben, der Apostroph ('), die Tilde (~) und zum Gruppieren die geschweiften Klammern,
- die Akzent-Befehle `\", \' , \. , \=, \^, \' , \~, \b, \c, \d, \H, \k, \r, \t, \u, \v` und
- die Sonderbuchstaben `\AE (\AE), \ae (\ae), \DH (\DH), \dh (\dh), \DJ (\DJ), \dj (\dj), \i (\i), \j (\j), \L (\L), \l (\l), \NG (\NG), \ng (\ng), \OE (\OE), \oe (\oe), \O (\O), \o (\o), \SS (\SS), \ss (\ss), \TH (\TH), \th (\th)`.

Die Sonderbuchstaben sind im Standardfont von $\text{\LaTeX}$ gesetzt, da bei dem im Buch verwendeten PostScript-Font die Zeichen `\j`, `\NG` und `\ng` fehlen.

Die im Paket `babel` zur Verfügung gestellten Abkürzungen etwa für Umlaute und Eszett können ebenfalls verwendet werden. Dabei ist aber zu beachten, dass `babel` für die in der Bibliografie verwendeten Sprachen in jedem Dokument geladen sein muss, das die Bibliografie verwendet. Wenn die Bibliografiedatei zur Dokumentation formatiert werden soll, muss sie ebenfalls das Paket `babel` laden. Das ist keine wesentliche Einschränkung, da es bei Einträgen in verschiedenen Sprachen ohnehin in der Regel erforderlich ist, `babel` mit diesen Sprachen zu laden.

Bei Vorgabe einer Zeichenkodierung mit dem Paket `inputenc` können auch Sonderzeichen wie Umlaute und Eszett in den Einträgen über die Tastatur eingegeben werden. Dazu müssen jedoch die Kodierungen von Dokument und Bibliografie übereinstimmen. Da diese Forderung die Weitergabe der Bibliografiedateien erschwert und eine Verwendung in Dokumenten mit anderer Zeichenkodierung verhindert, ist von der Verwendung kodierungsabhängiger Zeichen aber abzuraten.

Gelegentlich erfordert die Eingabe von Namen auch spezielle Akzente oder Symbole, die nicht vordefiniert sind. Auch für diese Fälle stehen Befehle zur Verfügung, um Akzente und Symbole für Namen zu definieren:

```
\DeclareNameAccent{<Akzent>}
```

definiert ein als Akzent zu verwendendes Zeichen `<Akzent>`. Dieser Befehl wird auch von `amsrefs` selbst zur Definition der bekannten Akzente verwendet:

```
\DeclareNameAccent{"}
```

Bei der Definition spezieller Symbole für Namen ist sowohl das Symbol bei Großschreibung anzugeben als auch das bei Kleinschreibung:

```
\DeclareNameSymbol{<Groß>}{<Klein>}
```

Die Argumente *<Groß>* und *<Klein>* müssen jeweils aus einem einzelnen Zeichen oder $\text{\LaTeX}$ -Befehl bestehen. Beispielsweise können auf diese Weise die deutschen Umlaute und Eszett definiert werden:

```
\DeclareNameSymbol{ä}{ä}
\DeclareNameSymbol{ö}{ö}
\DeclareNameSymbol{ü}{ü}
\DeclareNameSymbol{\ss}{ß}
```

13.2.2 Literaturverzeichnisse mit $\text{BIB}\text{\TeX}$

Literaturverzeichnisse können mit $\text{BIB}\text{\TeX}$ wie üblich aufgebaut werden. Einziger Unterschied ist, dass der `\bibliographystyle`-Befehl fehlt, da `amsrefs` automatisch einen zur Verwendung mit `amsrefs` geeigneten Bibliografiestil verlangt. Wie gewohnt wird das Literaturverzeichnis mit dem Befehl

```
\bibliography{<bib-Datei>}[,bib-Datei]...
```

gesetzt. $\text{BIB}\text{\TeX}$ erstellt aus der als Argument angegebenen `.bib`-Datei eine `.bbl`-Datei, die alle von `amsrefs` benötigten Befehle zum Setzen des Literaturverzeichnisses enthält, wie sie in den folgenden Abschnitten beschrieben sind. Die erzeugte `.bbl`-Datei entspricht dabei vollständig dem Inhalt der `document`-Umgebung einer `.ltb`-Datei und kann durch Hinzufügen von

```
\documentclass{article}
\usepackage{amsrefs}
\begin{document}
  <Inhalt der bbl-Datei>
\end{document}
```

leicht in eine vollständige `.ltb`-Datei umgewandelt werden, die sich zur Dokumentation des Inhalts mit $\text{\LaTeX}$ formatieren lässt.

13.2.3 Befehle zum Aufbau von Literaturverzeichnissen

Im Unterschied zu $\text{BIB}\text{T}_{\text{X}}$ wird die Form des gesetzten Literaturverzeichnisses nicht durch einen Befehl zur Angabe des Bibliografiestils festgelegt, sondern über Optionen beim Laden des Pakets. Abweichungen von den vorgesehenen Stilen lassen sich durch Laden zusätzlicher Pakete realisieren, die sicher im Laufe der Zeit zur Verfügung stehen werden.

Abweichend strukturiert ist auch das Literaturverzeichnis selbst, das bei manueller Eingabe oder bei $\text{BIB}\text{T}_{\text{X}}$ in einer Umgebung `thebibliography` eingebettet ist. An die Stelle von `thebibliography` treten zwei Umgebungen: Die äußere setzt die Überschrift und kann auch einleitenden Text enthalten, die innere enthält die eigentliche Literaturliste. Die innere Umgebung kann gegebenenfalls auch unabhängig von der äußeren verwendet werden, wenn eine Überschrift nicht erwünscht ist, etwa um die Überschrift und die Einleitung selbst zu gestalten. Gegebenenfalls kann die Umgebung auch mehrfach verwendet werden, beispielsweise für kapitelweise Literaturverzeichnisse.

`amsrefs` erzeugt bei Übernahme von Literaturdaten aus einer `.ltb`-Datei ebenfalls eine `.bbl`-Datei. Diese unterscheidet sich allerdings von einer mit Hilfe von $\text{BIB}\text{T}_{\text{X}}$ erzeugten dadurch, dass sie nur `\bib`-Befehle enthält, nicht aber die Listenumgebung und auch keinen Titel.

13.2.3.1 Der Titel

In vielen Fällen werden Literaturverzeichnisse als Kapitel oder eigener Abschnitt gesetzt, abhängig von der Dokumentklasse. `amsrefs` stellt dazu drei Umgebungen zur Verfügung, eine generische, die abhängig von der Dokumentklasse über den Überschriftenlevel entscheidet, sowie eine zum Setzen des Literaturverzeichnisses als Kapitel und eine zum Satz als Abschnitt. Die Umgebungen haben die folgende Form:

```
\begin{bibdiv}[<Überschrift>]
...
\end{bibdiv}
```

```
\begin{bibsection}[<Überschrift>]
...
\end{bibsection}
```

```
\begin{bibchapter}[<Überschrift>]
...
\end{bibchapter}
```

`bibdiv` ist die generische Umgebung und verhält sich je nach Dokumentklasse wie eine der beiden anderen. Aus den Namen der Umgebungen ist ersichtlich, dass `bibsection` das Literaturverzeichnis mit einem `\section`-Befehl setzt und `bibchapter` mit einem `\chapter`-Befehl. Das optionale Argument `<Überschrift>` erlaubt es, den voreingestellten Titel des Literaturverzeichnisses – der aktuelle Wert der auch von `abel` unterstützten Befehle `\bibname` bei `bibchapter` und `\refname` bei `bibsection` – zu überschreiben. Alle Umgebungen können beliebigen einleitenden Text enthalten.

13.2.3.2 Die Literaturliste

Für die Literaturliste verwendet `amsrefs` die Umgebung `biblist`. Sie hat die folgende Form:

```
\begin{biblist}[<Vereinbarungen>]
  \bib...
\end{biblist}
```

Die Umgebung `biblist` stellt den für die Zitiermarke benötigten Platz selbst bei der ersten Formatierung fest und reserviert ihn ab der zweiten Formatierung. Das vorgeschriebene Argument bei `thebibliography` fehlt daher. Statt dessen erlaubt das optionale Argument `<Vereinbarungen>` das Verändern der Voreinstellungen.

Handelt es sich bei den Zitiermarken um Nummern, beginnt die voreingestellte Nummerierung bei 1. Besteht nun die Literaturliste aus mehreren Teilen, kann die Variante

```
\begin{biblist*}[<Vereinbarungen>]
  \bib...
\end{biblist*}
```

verwendet werden, um im Anschluss an die vorhergehende Literaturliste weiter zu nummerieren.

In der Form unterscheiden sich die vorgesehenen Varianten der Literaturliste nur in den verwendeten Zitiermarken, die durch Angabe einer der Optionen `alphabetic`, `shortalphabetic`, `author-year`, `y2k` oder

der voreingestellten Option `numeric` festgelegt werden. Die Literaturliste wird in der Schriftgröße `\footnotesize` gesetzt. Alternative Stile, die als Paket geladen werden, stehen noch nicht zur Verfügung. Sobald das Paket auch außerhalb der Veröffentlichungen der $\mathcal{A}\mathcal{M}\mathcal{S}$ häufiger eingesetzt wird, werden sicher solche Pakete entwickelt. Bis dahin können Änderungen mit den in Abschnitt 13.2.8 beschriebenen Befehlen vorgenommen werden, wenn Festlegungen im optionalen Argument `<Vereinbarungen>` nicht ausreichen. Ein direkter Einfluss kann auf die Einträge nicht genommen werden, da eine mit Formatierungsangaben versehene Literaturliste, wie sie von $\text{\LaTeX}$ erstellt wird, nur intern aufgebaut wird und nicht zugänglich ist.

Sehr einfach ist es, mit Hilfe des optionalen Arguments die Schriftgröße zu ändern und die Literaturliste kursiv in der normalen Schriftgröße zu setzen:

```
\begin{biblist}[\itshape\normalsize]
...
\end{biblist}
```

Für eine nummerierte Literaturliste wird der Zähler `bib` verwendet. In `<Vereinbarungen>` ist es daher leicht möglich, die Anfangsnummer vorzugeben. Soll eine Literaturliste beispielsweise mit der Nummer 0 statt 1 beginnen, kann das in der folgenden Form möglich:

```
\begin{biblist}[\setcounter{bib}{-1}]
...
\end{biblist}
```

Natürlich können auch in `amsrefs` definierte Befehle in `<Vereinbarungen>` angegeben werden. Beispielsweise legt der Befehl

```
\resetbiblist{<Marke>}
```

den Platz für die Zitiermarke in der Literaturliste reservierten Platz so fest, als wäre `<Marke>` die längste vorkommende Zitiermarke. Beansprucht eine Zitiermarke in der Liste mehr Platz, ragt sie in den Eintrag hinein, wie es in einer `description`-Liste üblich ist. Der Befehl kann etwa verwendet werden, um ein aus mehreren einzelnen `biblist`-Umgebungen bestehendes Literaturverzeichnis einheitlich einzurücken, da normalerweise `amsrefs` die Einrückung ab der zweiten Formatierung selbst korrekt setzt.

```
\begin{biblist*}[\resetbiblist{999}]
...
\end{biblist*}
```

Die Literaturdaten können entweder mit einem Befehl aus `.ltb`-Dateien übernommen oder manuell mit dem `\bib`-Befehl eingetragen werden.

13.2.3.3 Übernahme gespeicherter Literaturdaten

Neben der Möglichkeit zur Übernahme von $\text{BIB}\text{T}\text{E}\text{X}$ Literaturdaten aus ‚ $\text{BIB}\text{T}\text{E}\text{X}$ -Datenbanken‘ können auch speziell für `amsrefs` angelegte Literaturdaten aus `.ltb`-Dateien übernommen werden. Diese `.ltb`-Dateien sind gewöhnliche $\text{L}\text{T}\text{E}\text{X}$ -Dokumente, die das Paket `amsrefs` laden und in der Regel lediglich ein Literaturverzeichnis enthalten, das mit den in diesem Abschnitt beschriebenen Befehlen aufgebaut wird. Aus der `.ltb`-Datei werden die zitierten Einträge (bei jedem Formatieren neu) in eine `.bbl`-Datei übertragen, die dann eingebunden wird.

Eine Sortierung wie bei $\text{BIB}\text{T}\text{E}\text{X}$ findet dabei nicht statt. Allerdings kann die `.bbl`-Datei sortiert und im Dokument an die Stelle des Befehls zum Übernehmen der Daten aus einer oder mehreren `.ltb`-Dateien gesetzt werden. Dafür bieten die `.ltb`-Dateien aber auch eine Reihe von Vorteilen gegenüber $\text{BIB}\text{T}\text{E}\text{X}$:

Erhaltung der Struktur

Ein wichtiger Vorteil ist, dass die `.ltb`-Datei oder das $\text{L}\text{T}\text{E}\text{X}$ -Dokument die vollständigen strukturierten Daten enthält. Diese Daten stehen beim Formatieren also komplett zur Verfügung und können beim Setzen berücksichtigt werden, wenn der Stil des Literaturverzeichnisses das vorsieht. Damit vereinfacht sich auch die Weitergabe von Dokumenten mit einem Literaturverzeichnis.

Strukturierte statt formatierte Daten

Die aus einer `.bib`- oder `.ltb`-Datei extrahierten Daten enthalten keine Formatierungsangaben. Beim Wechsel des Stils des Literaturverzeichnisses müssen sie daher nicht neu extrahiert werden. Das vereinfacht vor allem die Arbeit mit $\text{BIB}\text{T}\text{E}\text{X}$.

Keine Änderung der Schreibweise

Im Unterschied zu $\text{BIB}\text{T}\text{E}\text{X}$ ändert `amsrefs` die Groß-/Kleinschreibung von Titeln nicht. Insbesondere bei deutschen Titeln besteht daher keine Notwendigkeit, große Anfangsbuchstaben von Wörtern in geschweifte Klammern zu setzen, um bei Zeitschriftenartikeln die von $\text{BIB}\text{T}\text{E}\text{X}$ vorgenommene Änderung in kleine Buchstaben zu verhindern.

Selbstdokumentierende `.ltb`-Dateien

Wie oben beschrieben handelt es sich bei den `.ltb`-Dateien um normale $\text{L}\text{T}\text{E}\text{X}$ -Dokumente, die außer der Bibliografie in der Regel kei-

nen Text enthalten. Das Formatieren mit $\LaTeX$ setzt das Literaturverzeichnis und dokumentiert damit den Inhalt der `.ltb`-Datei.

Stiländerungen mit $\LaTeX$ -Kenntnissen

Im Unterschied zu $\BIBTeX$ ist zum Ändern des Stils des Literaturverzeichnisses nicht das Erlernen einer speziellen Sprache erforderlich. Die Definition eines Stils erfolgt nur mit normalen $\LaTeX$ -Befehlen und im Paket `amsrefs` definierten Befehlen. Neue Stile können dann in Form von $\LaTeX$ -Paketen zur Verfügung gestellt werden.

Feste Form von Namen und Daten

Namen und Daten werden bei `amsrefs` in einer fest vorgegebenen Form angegeben. Bei Daten muss das beim Feld `date` auf Seite 461 beschriebene ISO 8601 Format verwendet werden.

Namen müssen in der Form

`<Name>, <Vorname (n)>[, <Zusätze>]`

angegeben werden, wobei `<Zusätze>` Angaben wie `Jr.` enthält. Diese fest vorgegebene Form vermeidet Fehlinterpretationen, wie sie bei $\BIBTeX$, das keine feste Form vorschreibt, gelegentlich vorkommen.

Einheitliche Form der Literaturdaten

Manuell zusammengestellte Literaturverzeichnisse haben die gleiche Form wie die Angaben in einer `.ltb`-Datei, da die Einträge nur die Struktur enthalten, nicht aber eine spezielle Formatierung. Die Aufbereitung der Informationen zum Satz des Literaturverzeichnisses erfolgt beim Formatieren.

Für Dokumente mit Literatur, die voraussichtlich nicht erneut zitiert werden muss – etwa Studien-, Magister- oder Diplomarbeiten – eignen sich manuell erstellte Literaturverzeichnisse mit `amsrefs` am besten. Sollte sich später herausstellen, dass Einträge doch häufiger zitiert werden, lassen sich diese Einträge leicht ohne Änderungen in eine `.ltb`-Datei kopieren.

Arbeitet man hingegen längere Zeit in einem festen Fachgebiet, werden zumindest einige grundlegende Veröffentlichungen immer wieder zitiert. In solchen Fällen empfiehlt sich die Anlage von `.ltb`-Dateien, aus denen die aktuell zitierten mit einem einfachen Befehl extrahiert werden können. So müssen die Daten nicht immer wieder neu eingegeben oder kopiert werden. Der Befehl zum Extrahieren und Einfügen der Daten hat die folgende Form:

```
\bibselect{<ltb-Datei>}[,ltb-Datei]...
```

Wie beim `\bibliography`-Befehl können mehrere `.ltb`-Dateien angegeben werden, aus denen die zitierten Veröffentlichungen übernommen werden. Alternativ kann der `\bibselect`-Befehl auch mehrfach angegeben werden.

Um alle Veröffentlichungen aus `.ltb`-Dateien zu übernehmen, kann der Befehl

```
\bibselect*{<ltb-Datei>}[.ltb-Datei]...
```

verwendet werden. Zusammen mit dem `\bibselect`-Befehl erlaubt er die Auswahl, welche `.ltb`-Dateien komplett und aus welchen nur die zitierten Veröffentlichungen übernommen werden sollen. Der ebenfalls mögliche Befehl

```
\nocite{*}
```

wirkt im Unterschied dazu auf alle folgenden `\bibselect`-Befehle.

13.2.3.4 Angabe der Literaturdaten mit dem `bib`-Befehl

Die Daten einer Veröffentlichung werden mit dem Befehl `\bib` direkt in das Literaturverzeichnis oder in eine $\mathcal{A}\mathcal{M}\mathcal{S}$ -Literaturdatei (`.ltb`-Datei) geschrieben, die selbst ein ‚normales‘ $\text{\LaTeX}$ -Dokument ist, das lediglich aus einem Literaturverzeichnis besteht. Der Befehl hat dabei die folgende allgemeine Form:

```
\bib{<Marke>}{<Typ>}{
  <Feld>={<Wert>}[ ,
  <Feld>={<Wert>}] ...
}
```

`<Marke>` wird in Zitierbefehlen zum Verweisen auf den Literatureintrag verwendet. `<Typ>` ist einer der in 13.2.4 beschriebenen Eintragstypen. Die Daten des Literaturverweises werden als `<Wert>` den Feldern `<Feld>` zugewiesen, wobei `<Feld>` einer der in 13.2.4.1 beschriebenen Feldnamen ist. Bei zusammengesetzten Feldern ist der Wert wiederum eine Liste wie das dritte Argument des `\bib`-Befehls. Im Unterschied zu $\text{\BIBTeX}$ dürfen die geschweiften Klammern um den Wert nicht fehlen.

Bei Verwendung des `\bib`-Befehls ist zu beachten, dass die *Formatierung der Eingabe* eine wesentliche Rolle spielt:

- Der Befehl muss in einer neuen Zeile beginnen.

- Die ersten beiden Argument `<Marke>` und `<Typ>` müssen in der ersten Zeile stehen.
- Die öffnende Klammer des dritten Arguments muss ebenfalls in der ersten Zeile stehen.

Die erste Zeile muss also genau die angegebene Form haben. Andernfalls ist es möglich, dass amsrefs den Eintrag nicht interpretieren kann und sogar die Formatierung des Dokuments abgebrochen wird. Es ist also wesentlich, sich an die beschriebene Form zu halten.

Auch der `\bib`-Befehl hat eine Sonderform. Diese Sonderform erlaubt es, gemeinsame Daten mehrerer einzelner Literaturangaben zusammenzufassen.

```
\bib*{<Marke>}{<Typ>}{
  <Feld>={<Wert>}[,
  <Feld>={<Wert>}] ...
}
```

Die Argumente haben die gleiche Bedeutung wie beim `\bib`-Befehl. Mit `\bib*` festgelegte Angaben können in anderen Literatureinträgen über das Feld `xref` übernommen werden, was bei ‚normalen‘ Literatureinträgen nicht möglich ist. Auf der anderen Seite setzt `\bib*` nicht selbst einen Eintrag in das Literaturverzeichnis.

13.2.3.5 Gemeinsamkeiten und Unterschiede im Vergleich mit **BIB_{TeX}**

amsrefs und BIB_{TeX} verwenden die gleichen Feldnamen für die Literaturdaten. Auch in der Form, in der Werte für die Felder angegeben werden, gibt es kaum Unterschiede. Mit gewissen Einschränkungen können diese Angaben unverändert von einem zum anderen übernommen werden.

Bereits optisch unterscheiden sich die Einträge für amsrefs und BIB_{TeX} sehr deutlich in der Grundform eines Eintrags:

amsrefs

```
\bib{<Marke>}{<Typ>}{
  <Felddefinitionen>
}
```

BIB_{TeX}

```
@<Typ>{<Marke>,
  <Felddefinitionen>
}
```

Bei amsrefs werden also `<Marke>` und `<Typ>` beide – in dieser Reihenfolge – als Argument angegeben, und das dritte Argument enthält die

<Felddefinitionen>. Bei $\text{BIB}\text{T}_{\text{E}}\text{X}$ ist *<Typ>* Teil der Kennung, mit der ein neuer Eintrag begonnen wird, und *<Marke>* ist die erste Angabe im einzigen Argument, dem – nach einem trennenden Komma – die *<Felddefinitionen>* folgen.

Die übrigen Unterschiede sind – je nach der gewählten Form – nicht so auffällig. In einigen Fällen erlaubt $\text{BIB}\text{T}_{\text{E}}\text{X}$ die von amsrefs vorgeschriebene Form, lässt aber auch andere Formen zu.

Schreibweise der Feldnamen

$\text{BIB}\text{T}_{\text{E}}\text{X}$ unterscheidet bei Feldnamen und Eintragsarten Groß- und Kleinbuchstaben nicht. In amsrefs werden jedoch aus den Feldnamen und Typbezeichnungen Teile von Befehlen, und da $\text{T}_{\text{E}}\text{X}$ bei Befehlen Groß- und Kleinbuchstaben unterscheidet, müssen die Feldnamen und Eintragsarten wie angegeben in Kleinbuchstaben geschrieben werden.

Vorgeschriebenes Einklammern der Feldinhalte in geschweifte Klammern

Bei $\text{BIB}\text{T}_{\text{E}}\text{X}$ sind geschweifte Klammern nur unter bestimmten Bedingungen zwingend erforderlich; bei einfachen Angaben wie Zahlen können sie entfallen, und, in anderen Fällen wie einfachen Texten, können statt der Klammern doppelte Anführungszeichen verwendet werden. amsrefs verlangt hingegen, dass alle Feldinhalte ohne Ausnahme in geschweifte Klammern eingeschlossen sind, und meldet andernfalls einen Fehler.

Wiederholbare Felder

$\text{BIB}\text{T}_{\text{E}}\text{X}$ kennt keine wiederholbaren Felder. Sind mehrfache Angaben erforderlich wie beispielsweise bei mehreren Autoren, müssen die Angaben komplett in dem einen Feld untergebracht werden. Dadurch entstehen allerdings Probleme, wenn die Angaben analysiert werden müssen, etwa im Fall von Namen, bei denen die Vornamen abgekürzt werden sollen. amsrefs vermeidet diese Probleme und lässt es zu, dass bestimmte Felder mehrfach angegeben werden wie zum Beispiel das Feld für einen Autor oder Editor. Bei diesen wiederholbaren Feldern sollte als Wert jeweils nur eine einzelne Angabe gemacht werden, beispielsweise der Name eines einzelnen Autors. Für weitere Einzelangaben sollte das Feld – sooft wie erforderlich – wiederholt werden. amsrefs fasst die Einzelangaben zu einer geeigneten Liste zusammen.

Namen

$\text{BIB}\text{T}_{\text{E}}\text{X}$ lässt bei der Angabe von Namen viele Freiheiten. Dadurch ist es allerdings gelegentlich schwierig oder in besonderen Fällen sogar unmöglich, den Namen zu analysieren und in Vornamen, Nachnamen und Zusätze zu zerlegen. Das kann dann zu falschen Abkürzungen oder auch irreführenden automatisch generierten Zitiermar-

ken führen. Durch Angabe des Namens in einer für ihn geeigneten Form lassen sich diese Schwierigkeiten vermeiden, aber es ist gelegentlich schwierig vorherzusehen, welche Form das ist. amsrefs vermeidet diese, indem es vorschreibt, dass Namen grundsätzlich in der Form `<Nachname>`, `<Vorname (n)>`[, `<Zusätze>`] angegeben werden, also beispielsweise

```
author={Jones, John Paul}
author={van Beethoven, Ludwig}
author={Ford, Henry, Jr.}
```

Im Satz erscheinen die Namen in der üblichen Reihenfolge `<Vorname (n)>` `<Nachname>` gegebenenfalls gefolgt von den Zusätzen nach einem trennenden Komma. In einigen Ländern – etwa in China – ist diese Reihenfolge allerdings nicht üblich, und für solche Fälle sieht amsrefs die Angabe eines Feldattributs vor, um den Namen in der Reihenfolge `<Nachname>` `<Vorname (n)>` zu setzen.

Datumsangaben

Auch bei der Angabe von Daten lässt L^AT_EX unterschiedliche Formen zu. Zur Vermeidung von Schwierigkeiten etwa bei Verwendung von Monatsnamen schreibt amsrefs vor, dass Daten im ISO 8601 Format `jjjj-mm-tt` anzugeben sind.

Automatische Anpassung von Anfangsbuchstaben bei Titeln

L^AT_EX passt die Groß-/Kleinschreibung in Titeln automatisch an die amerikanischen Konventionen an, nach denen bei Zeitschriftenartikeln außer dem ersten Wort alle anderen klein geschrieben werden, während bei Büchern alle Worte mit Ausnahme von Artikeln und sehr kurzen (Füll-)Wörtern mit großem Anfangsbuchstaben geschrieben werden. Die automatische Anpassung – etwa bei deutschen Titeln erforderlich – lässt sich nur durch besondere Vorkehrungen verhindern. Da diese automatische Anpassung selbst in mathematischen Formeln durchgeführt wird, nimmt amsrefs *keine* automatische Anpassung vor, sondern die vorgegebene Groß-/Kleinschreibung wird beibehalten. Bei deutschen Titeln ist das sicher ein Vorteil. Bei englischen Titeln obliegt es dem Autor, die gewünschte oder verlangte Groß-/Kleinschreibung bei der Eingabe des Titels zu berücksichtigen.

Spezielle Zeichen

Sonderzeichen wie Umlaute oder Eszett können bei amsrefs in der gleichen Weise wie im Dokument eingegeben werden. Der Einschluss in geschweifte Klammern wie bei L^AT_EX kann entfallen. Andererseits können geschweifte Klammern zur Gruppierung von Buchstaben verwendet werden, beispielsweise wenn ein einzelner (Anfangs-) Buchstabe in einem fremden Alphabet bei der Transskription in mehrere Buchstaben umgesetzt wird oder spezielle Schreibweisen erforderlich sind:

```
author={Gagarin, {Yu}ri}
author={Katzenbach, Nicholas {deB}elleville}
```

führen ebenso zu den abgekürzten Vornamen *Yu.* und *deB.* wie die Angaben

```
author={Gagarin, Yu.}
author={Katzenbach, N. deB.}
```

mit von vorneherein abgekürzten Vornamen (die Angabe des Punkts verhindert eine weitere Abkürzung).

13.2.4 Eintragstypen

Ebenso wie $\text{BIB}\text{T}_{\text{E}}\text{X}$ kennt amsrefs mehrere Eintragstypen, etwa für Bücher, Zeitschriftenartikel oder Dissertationen. Die Eintragstypen von $\text{BIB}\text{T}_{\text{E}}\text{X}$ werden alle unterstützt, aber es sind nur vier Grundtypen definiert. Die übrigen Eintragstypen sind Synonyme für einen dieser Grundtypen. Die verringerte Anzahl der Typen wird aber mehr als ausgeglichen durch zusätzliche Möglichkeiten bei den Feldern der Einträge. Die folgende Liste beschreibt die Grundtypen.

article

Für kürzere Veröffentlichungen, beispielsweise Artikel in Zeitschriften oder Sammelbänden oder Veröffentlichungen im WWW.

book Für umfangreichere Veröffentlichungen eines Autors oder mehrerer Autoren, vor allem in Buchform.

report

Für technische Handbücher oder ähnliche Veröffentlichungen.

Ähnlich wie **article**, aber in der Regel veröffentlicht und vertrieben von Organisationen wie Universitäten oder Firmen, deren Geschäftszweck nicht im Veröffentlichen besteht (wie bei Verlagen der Fall).

thesis

Für wissenschaftliche Arbeiten mit dem Ziel, einen Abschluss zu erreichen, etwa Diplom- Magister- oder Habilitation.

Neben diesen Grundtypen gibt es weitere Eintragstypen, die vor allem mit Rücksicht auf die in $\text{BIB}\text{T}_{\text{E}}\text{X}$ verwendeten Typen zur Verfügung gestellt werden. Es handelt sich dabei um Synonyme für Grundtypen. Sie können gegebenenfalls verwendet werden, um den Typ von Veröffentlichungen genauer zu kennzeichnen. Andererseits ist es möglich, Bibliografiestile

zu entwickeln, die diese Typen abweichend von den Grundtypen umsetzen. Die feinere Differenzierung der Eintragstypen kann also durchaus sinnvoll sein, da bei zunehmender Verwendung des Pakets voraussichtlich auch andere Stile zur Verfügung stehen werden.

amsrefs behandelt die folgenden Eintragstypen als Synonyme:

Synonyme für book

collection	Eine Sammlung von Artikeln eines Autors oder mehrerer Autoren
manual	Für Handbücher
misc	Kurzform für <code>miscellaneous</code>
miscellaneous	Veröffentlichungen, die anderen Eintragstypen nicht zuzuordnen sind.
periodical	Regelmäßig erscheinende Publikationen wie Tages- oder Wochenzeitungen sowie Zeitschriften.
proceedings	Sammelband mit Veröffentlichungen von Konferenzbeiträgen.
unpublished	(Noch) unveröffentlichte Artikel oder Dokumente.

Synonyme für report

techreport	Technische Handbücher.
-------------------	------------------------

Für die Eintragstypen `article` und `thesis` gibt es keine Synonyme.

13.2.4.1 Die Felder im bib-Befehl

amsrefs unterscheidet verschiedene Typen von Feldern. Da sind zunächst die *einfachen Felder*, die in einem Eintrag nur einmal verwendet werden können und einen einzelnen Wert enthalten. Dieser Typ ist am Weitesten verbreitet.

Ein weiterer Typ sind *wiederholbare Felder*, die ebenfalls nur einen einzelnen Wert haben können. Die wichtigsten Felder dieses Typs sind `author` zur Angabe eines Autors und `editor` für einen Herausgeber. Im Unterschied zu `LATEX` sollten diese Felder keine Listen enthalten, da es einfacher ist, die Daten zu bearbeiten und daraus eine Liste aufzubauen als eine Liste zu analysieren und gegebenenfalls die Elemente der Liste zu bearbeiten.

Der dritte Typ sind *zusammengesetzte Felder*. Hier handelt es sich um komplexe Angaben, die ihrerseits andere Felder enthalten, beispielsweise die Angaben zu einer Konferenz, in deren Umfeld ein Artikel veröffentlicht wurde oder Angaben zum Buch, in dem der Artikel neben anderen zu finden ist. Ein anderes Beispiel sind Artikel, die in mehreren Teilen veröffentlicht wurden. Die Angaben zu den einzelnen Teilen werden dann in jeweils einem zusammengesetzten Feld erfasst. Mit einer Ausnahme können zusammengesetzte Felder nur bei bestimmten Eintragstypen verwendet werden.

Auch einige zusammengesetzte Felder können mehrfach vorhanden sein. Da es sich aber nur um wenige Felder handelt, wird nicht nach dieser Eigenschaft unterschieden. Statt dessen wird bei der Beschreibung darauf hingewiesen, dass das Feld wiederholt werden kann.

Die folgende Tabelle führt alle Felder – gruppiert in Feldtypen – zur Übersicht auf. Anschließend werden die einzelnen Felder nach Typ getrennt beschrieben.

Einfache Felder		Wiederholbare Felder
address [†]	art	author
booktitle	place [†]	editor
date [†]	publisher	isbn
doi	school [†]	issn
edition	series	review
eprint	setup	translator
hyphenation	status	
institution [†]	subtitle	
journal	title	Zusammengesetzte Felder
label	transition	book
language	type	conference
note	url	contribution
number	volume	partial
organization [†]	xref	reprint
pages	year [†]	translation

[†] Die so markierten Felder sind Synonyme, und zwar `institution` und `school` für `organization`, `place` für `address` sowie `year` für `date`.

Zwei der hier aufgeführten Felder (`language` und `hyphenation`) legen die Sprache und die für den Eintrag verwendeten Trennmuster fest. `language` wird auch als Information über die Sprache der Veröffentlichung in das Literaturverzeichnis aufgenommen. Bei Angabe einer babel bekannten Sprache (im Deutschen in der Regel nicht passend) wird der

zugewiesene Wert auch zur Wahl des Trennmusters für den Eintrag verwendet. `hyphenation` legt nur das Trennmuster fest.

Die Trennmuster werden nach den folgenden Regeln gewählt:

1. Laut Angabe im Feld `hyphenation`.
2. Entsprechend dem ersten Wort im Feld `language`, wenn es eine babel bekannte Sprachbezeichnung ist.
3. Entsprechend der aktuellen Sprache (direkt) vor Ausführung des `\bib`-Befehls.
4. Entsprechend dem aktuellen Trennmuster des Dokuments.

Die Felder `label`, `setup` und `xref` helfen bei der Organisation eines Eintrags. `label` erlaubt die Angabe einer Zitiermarke bei Stilen, die alphabetische Zitiermarken verwenden. Im Feld `setup` können Formatierungsbefehle angegeben werden, die auf den gesamten Eintrag angewendet werden. Durch Angabe vom `\bfseries` könnte man beispielsweise besonders wichtige Veröffentlichungen hervorheben. `xref` verweist auf einen mit `\bib*` definierten Eintrag, dessen Informationen in den aktuellen Eintrag übernommen werden sollen.

Die übrigen Felder enthalten Informationen zu der durch den Eintrag beschriebenen Veröffentlichung. Die folgende Liste beschreibt die Felder, die von `amsrefs` unterstützt werden. Auch die bereits erwähnten Felder werden nochmals behandelt.

Einfache Felder

address

Adresse – häufig nur der Ort – des Verlags oder der veröffentlichenden Organisation und innerhalb des zusammengesetzten Felds `conference` Adresse der Konferenz.

booktitle

Anderer Name für das Feld `title`, der von `LATEX` verwendet wird.

Beim Aufbau neuer Literaturdaten für `amsrefs` sollte das Feld `title` verwendet werden.

date Datum der Veröffentlichung im ISO 8601 Format, also in der Form `jjjj-mm-tt` (`j` für die Ziffern der Jahreszahl, `m` für die Ziffern des Monats und `t` für die des Tages). Tag und Monat können bei der Datumsangabe fehlen.

Bei der Angabe des Monats sind außer den Zahlen für die zwölf Monate auch die Angaben 13 für den Winter, 14 für das Frühjahr, 15 für den Sommer und 16 für den Herbst möglich. In diesem Fällen muss die Angabe eines Tages fehlen, da sie andernfalls – wie bei einem ‚normalen‘ Datum – gesetzt wird.

In der dieser Beschreibung zu Grunde liegenden Version sind die Monatsnamen und Jahreszeiten im Paket fest in Englisch vorgegeben.

doi ‚Digital Object Identifier‘, eine nur vom Eintragsartyp `article` berücksichtigtes Feld.

Bei Angaben in diesem Feld wird der Text ‚DOI *<Wert>*‘ gesetzt. Enthält das Feld `volume` Angaben, wird ‚... (to appear in print)‘ hinzugefügt. Zusätzlich wird der normalerweise in runden Klammern gesetzte Wert des Felds `date` in der Form ‚posted on *<Datum>*‘ in das Literaturverzeichnis übernommen.

edition

Auflage oder Edition bei Büchern.

Handelt es sich um eine Zahlangabe, wird die Zahl vom Befehl `\CardinalNumber` umgewandelt (etwa `first` oder `6th`), und es wird der Text in `\editiontext` (voreingestellt: `ed.`) angehängt.

In der dieser Beschreibung zu Grunde liegenden Version sind die Editionsangaben im Paket fest in Englisch vorgegeben.

Andere Angaben werden gesetzt wie angegeben.

eprint

Informationen zu einer elektronischen Vorveröffentlichung wie eine URL.

Angaben im Feld `eprint` werden in der Form ‚available at *<Wert>*‘ in das Literaturverzeichnis übernommen.

hyphenation

Sprache des Eintrags im Literaturverzeichnis im Sinne des Pakets `babel`, durch die die für den Eintrag zu verwendenden Trennmuster festgelegt werden.

Die Sprache für ein einzelnes Feld kann mit Hilfe des Attributs `language` festgelegt werden.

institution

Zur Kompatibilität mit $\text{BIB}\text{T}\text{E}\text{X}$ zugelassener alternativer Name für das Feld `organization`, dort vorgesehen für eine für die Veröffentlichung zuständige Institution wie etwa eine staatliche Einrichtung.

Beim Erfassen von Literaturdaten für amsrefs sollte der Feldname `organization` verwendet werden.

journal

Name der Zeitschrift, in der die Veröffentlichung erfolgte.

label

Vorgabe einer Zitiermarke, der im Fall der Optionen `alphabetic` oder `shortalphabetic` statt der automatisch generierten Zitiermarke verwendet wird. Damit lassen sich vom Standard abweichende Zitiermarken realisieren.

language

Sprache der Veröffentlichung, die auch im Literaturverzeichnis ausgegeben wird.

Die Sprache hat in der Regel keinen Einfluss auf das Verhalten von $\LaTeX$ und kann auch komplexere Angaben wie ‚Polnisch, mit englischer Zusammenfassung‘ enthalten.

note Anmerkungen zur Veröffentlichung, die auch im Literaturverzeichnis ausgegeben werden.

number

Nummer der Zeitschrift oder des Reports, in der die Veröffentlichung erfolgte.

Der Wert des Felds `number` wird bei einem Report in der Form ‚Technical Report *<Wert>*‘ und bei einem Artikel in der Form ‚no. *<Wert>*‘ in das Literaturverzeichnis übernommen.

organization

Für die Veröffentlichung zuständige Organisation, etwa eine Firma, ein Verein, ...

pages

Angabe der Seiten der Veröffentlichung.

part Angaben für längere Artikel in Zeitschriften, die auf getrennte Ausgaben verteilt veröffentlicht wurden.

place

Alternativer Name für das Feld `address`, der in früheren Versionen von amsrefs bevorzugt wurde. Aus Gründen der Kompatibilität mit $\BIBTeX$ wird nun allerdings doch der Name `address` bevorzugt.

Beim Erfassen von Literaturdaten sollte der Feldname `address` verwendet werden.

publisher

Name des Verlags.

school

Zur Kompatibilität mit BIB_{TeX} zugelassener alternativer Name für das Feld `organization`, dort vorgesehen für eine für die Veröffentlichung zuständige Institution wie etwa eine Universität.

Beim Erfassen von Literaturdaten für `amsrefs` sollte der Feldname `organization` verwendet werden.

series

Titel einer Buchreihe.

setup

Ein spezielles Feld, das verwendet werden kann, um – nach Übernahme aller Felder – vor Beginn der Bearbeitung des aktuellen `\bib`-Eintrags Befehle auszuführen, etwa um die Formatierung dieses speziellen Eintrags zu verändern. Es wäre beispielsweise möglich, durch Angabe von `\itshape` als Wert den aktuellen Eintrag durch Kursivschrift hervorzuheben. In der Regel wird man dieses Feld daher nur in einem `\bib`-Befehl in einem Dokument verwenden – gegebenenfalls nach Kopieren der `.bbl`-Datei in das Dokument.

status

Typischerweise Angaben wie ‚in Vorbereitung‘ oder ‚erscheint demnächst‘ im Fall von Zeitschriftenartikeln.

subtitle

Ein Untertitel einer Veröffentlichung, wie er typischerweise bei mehrteiligen Zeitschriftenartikeln oder gelegentlich bei mehrbändigen Werken für die einzelnen Teile verwendet wird.

title

Titel der Veröffentlichung.

transition

Ein Dummy-Feld, das beim Anpassen eines Bibliografiestils mit `\BibSpecs` verwendet wird, um die ‚unbedingte‘ Ausführung einer Aktion – etwa das Setzen eines zugehörigen Trennzeichens zwischen Literaturangaben – zu erzwingen, auch wenn die entsprechenden Angaben fehlen.

type Typisierungsangaben, die auch im Literaturverzeichnis ausgegeben werden, etwa ‚Dissertation‘ oder ‚Diplomarbeit‘. ‚Ph.D. Thesis‘ kann durch `p` und ‚Master’s Thesis‘ durch `m` abgekürzt werden.

Bei Verwendung im zusammengesetzten Feld `contribution` die Art des Beitrags, beispielsweise ‚Illustrationen‘.

url Angabe einer URL (engl. *Uniform Resource Locator*) für Verweise ins WWW, die zur Zeit noch nicht berücksichtigt wird. Der Name sollte aber bereits jetzt für die Zukunft benutzt werden.

volume

Nummer des Bandes bei mehrbändigen Werken oder Zeitschriften.

Angaben in den Feldern `book`, `booktitle` oder `conference` verhindern, dass die Angaben im Feld `volume` in das Literaturverzeichnis übernommen werden.

xref Verweis zu einem anderen `\bib`-Eintrag.

Alle Felder des Eintrags, auf den verwiesen wird, werden in die Daten des aktuellen Eintrags übernommen (kopiert). Bei einem Artikel mit einem Verweis auf einen Eintrag des Typs `collection` wird dessen Titel dem Feld `booktitle` zugewiesen.

year Anderer Name für das Feld `date`, der von `BIBTeX` für das Jahr der Veröffentlichung verwendet wird.

Beim Aufbau neuer Literaturdaten für `amsrefs` sollte das Feld `date` verwendet werden.

Wiederholbare Felder

author

Name eines Autors in der von `amsrefs` vorgeschriebenen Form (vgl. 13.2.1).

Da das Feld mehrfach verwendet werden kann, sollte es nur die Angaben zu einer einzigen Person enthalten.

editor

Name eines Herausgebers, der in der gleichen Form wie ein Autorenname angegeben werden sollte.

Da das Feld mehrfach verwendet werden kann, sollte es nur die Angaben zu einer einzigen Person enthalten.

isbn ISBN (engl. *International Standard Book Number*).

issn ISSN (engl. *International Standard Serial Number*), etwa bei Zeitschriften.

review

Nummer oder ein anderer Verweis auf eine Buchbesprechung, beispielsweise im ‚Zentralblatt der Mathematik‘ oder in ‚Mathematical Reviews‘.

translator

Name eines Übersetzers, der in der gleichen Form wie ein Autorenname angegeben werden sollte.

Da das Feld mehrfach verwendet werden kann, sollte es nur die Angaben zu einer einzigen Person enthalten.

Zusammengesetzte Felder

Wiederholbare und zusammengesetzte Felder behalten diese Eigenschaft auch bei Verwendung in zusammengesetzten Feldern.

book Angaben zu einem Buch, im dem einzelne Artikel zusammengefasst sind, etwa die Beiträge zu einer Konferenz bei Veröffentlichung in den ‚Proceedings‘. Das Feld soll die Unterscheidung zwischen einem Artikel und dem Buch, in dem er erschienen ist, einfacher machen. Es wird häufig in Verbindung mit dem Feld `conference` verwendet.

In diesem Feld zugelassene Felder: title, part, subtitle, edition, editor, translator, contribution, series, volume, publisher, organization, address, date, note

conference

Angaben zu einer Konferenz, in deren Umfeld die Veröffentlichung erfolgte.

In diesem Feld zugelassene Felder: title, address, date

contribution

Angaben zu Beiträgen, die nicht von Autoren, Herausgebern und Übersetzern kommen, beispielsweise Illustrationen.

In diesem Feld zugelassene Felder: type, author

Das Feld kann mehrfach verwendet werden. Es sollte nur die Angaben zu einem einzigen Typ von Beiträgen enthalten.

Die Angabe

```
contribution= {
  type={illustrations},
  author={Gorey, Edward}
}
```

führt unter Verwendung des voreingestellten Bibliografiestils zur Ausgabe von

with illustrations by Edward Gorey

In komplizierteren Fällen kann es erforderlich sein, zusätzlichen Text im `type`-Feld anzugeben, etwa `type={an appendix},.`

Obwohl das Paket `amsrefs` versucht, auch bei mehrfacher Angabe dieses Felds – das jeweils mehrere Autoren enthalten kann – eine adäquate Ausgabe zu erzeugen, kann es vorkommen, dass die Regeln, nach denen die Angaben gesetzt werden, nicht flexibel genug sind, um ein gutes Ergebnis zu erzielen. Als letzte Möglichkeit, die nur in solchen Fällen genutzt werden sollte, kann das Feld auch als einfaches Feld verwendet werden, das den zu setzenden Text komplett enthält:

```
contribution={mit beliebigen Angaben zu
              Beiträgen},
```

Der angegebene Text wird ohne Ergänzungen übernommen und setzt den folgenden Text:

mit beliebigen Angaben zu Beiträgen

partial

Dieses Feld ist speziell für Artikel (Typ `article`) vorgesehen, die in mehreren Teilen veröffentlicht werden.

In diesem Feld zugelassene Felder: `part`, `subtitle`, `contribution`, `journal`, `volume`, `date`, `number`, `pages`

Das Feld kann mehrfach verwendet werden. Es sollte nur die Angaben zu einem einzigen Teil der Veröffentlichung enthalten.

Angaben in den Feldern `book` oder `conference` verhindern, dass die Angaben im Feld `partial` in das Literaturverzeichnis übernommen werden.

reprint

Ebenfalls für Artikel (Typ `article`) zur Angabe einer Zweitveröffentlichung, üblicherweise in einem Buch. Daher kann es auch alle Angaben enthalten, die für das Feld `book` zugelassen sind.

In diesem Feld zugelassene Felder: `title`, `part`, `subtitle`, `edition`, `editor`, `translator`, `contribution`, `series`, `volume`, `publisher`, `organization`, `address`, `date`, `note`

translation

Angaben zu einer Übersetzung der Veröffentlichung.

Dieses ist das einzige zusammengesetzte Feld, das in jedem Bibliografietyp verwendet werden kann. Zugelassen sind alle Felder des aktuellen Eintrags.

Das BIBTeX -Feld `month` wird nicht unterstützt.

Einschränkungen bei der Angabe von Feldern gibt es nicht. Bei jedem Eintragsartyp können also sämtliche Felder angegeben sein. Beim Setzen der Bibliografie werden aber nur Felder berücksichtigt, die für den aktuellen Eintragsartyp vorgesehen sind. Die Angabe bestimmter Felder kann dazu führen, dass andere Felder, die andernfalls verwendet werden, nicht berücksichtigt werden.

Im Fall der Eintragsarten `book` und `report` und ihrer Synonyme muss ein Autor (`author`), Herausgeber (`editor`) oder Übersetzer (`translator`) angegeben werden. Bei `article` und `thesis` kann diese Angabe fehlen, und selbst dann werden Angaben in den Feldern `editor` und `translator` nicht berücksichtigt.

Die Felder `author`, `title` und `date` sowie `note`, `reprint`, `review`, `status`, `subtitle` und `translation` sind Standardfelder, die bei jeder Veröffentlichung gesetzt werden, sofern sie angegeben sind. Auch die bereits erwähnten Felder `language` und `hyphenation` für die Sprache und die Organisationsfelder `label`, `setup` und `xref` werden von allen Einträgen berücksichtigt. Durch Angabe des Felds `language` im zusammengesetzten Feld `translation` kann die Sprache angegeben werden, in die übersetzt wurde. Damit verwenden die `amsrefs`-Bibliografiestile bei allen Eintragsarten die folgenden Felder:

<code>author</code>	<code>date</code>	<code>hyphenation</code>	<code>label</code>	<code>language</code>
<code>note</code>	<code>reprint</code>	<code>review</code>	<code>setup</code>	<code>status</code>
<code>subtitle</code>	<code>title</code>	<code>translation</code>	<code>xref</code>	

Die Felder `isbn` und `issn` werden von den eingebauten Bibliografiestilen nicht berücksichtigt. Künftige Bibliografiestile werden diese Angaben aber wohl auch im Literaturverzeichnis setzen. Diese Felder können allerdings in selbst definierten Eintragsarten verwendet werden.

Aus Gründen der Kompatibilität zu BIBTeX kann statt `date` auch `year` verwendet werden. Wird als Wert von `year` ein vollständiges Datum wie bei `date` angegeben, gibt `amsrefs` eine Warnung aus.

Die verschiedenen Eintragsarten berücksichtigen neben diesen Standardfeldern unterschiedliche weitere Felder. Die folgende Liste beschreibt, welcher Eintragsart welche zusätzlichen Felder verwendet:

article	<code>book</code> , <code>booktitle</code> , <code>conference</code> , <code>contribution</code> , <code>doi</code> , <code>eprint</code> , <code>journal</code> , <code>number</code> , <code>pages</code> , <code>part</code> , <code>partial</code> , <code>volume</code>
book	<code>address</code> , <code>contribution</code> , <code>edition</code> , <code>editor</code> , <code>organization</code> , <code>part</code> , <code>publisher</code> , <code>series</code> , <code>translator</code> , <code>volume</code>

report	editor (bei fehlendem author), translator (bei fehlendem author und editor), address, contribution, edition, eprint, number, organization, part, series
thesis	address, eprint, organization, type

`place` sollte in einer früheren Version das in `BIBTeX` gebräuchliche `address` ersetzen. Diese Absicht wurde aus Kompatibilitätsgründen rückgängig gemacht, aber der damals eingeführte Name wird bis auf Weiteres als Synonym für `address` unterstützt. Die Eintragstypen `book`, `report` und `thesis` verwenden daher die Angaben im Feld `place` als Adresse, wenn das Feld `address` fehlt.

Aus Gründen der Kompatibilität mit `BIBTeX` werden für das Feld `organization` auch die Synonyme `institution` und `school` zugelassen. Ist `organization` nicht angegeben, wird `institution` verwendet. Nur wenn auch `institution` fehlt, wird `school` berücksichtigt.

13.2.5 Feldattribute

Obwohl `amsrefs` bei der Angabe von Namen eine feste Form vorgibt, werden Namen in bestimmten Fällen nicht korrekt umgesetzt, da sie nach den auch in Europa gültigen Konventionen umgesetzt werden. Chinesische Namen werden nach dieser Konvention jedoch nicht korrekt formatiert, da bei ihnen der Familienname vor den Vornamen angegeben wird. Da Vornamen von einigen Stilen abgekürzt werden, ist das Vertauschen der Vor- und Familiennamen nicht möglich. Das Problem lässt sich durch Angabe eines Attributes bei solchen Namen vermeiden, durch das `amsrefs` informiert wird, dass der Familienname an erster Stelle zu setzen ist.

Ein weiteres Problem ist die Sprache, in der die Felder gesetzt werden. Zwar kann die Sprache für einen Eintrag festgelegt werden, aber beispielsweise bei Übersetzungen sind bestimmte Felder in einer anderen Sprache anzugeben. Auch hier besteht die Lösung in der Angabe eines Attributs, das die Sprache des Felds festlegt.

Das Attribut zu einem Feld wird im Anschluss an seinen Wert in der folgenden Form angegeben:

```
<Feld>={<Wert>}*{<Attribut>={<Attributwert>}}
```

Vorgesehen sind die beiden oben erwähnten Attribute:

- inverted** kann bei allen Feldern angegeben werden, die Namen enthalten (`author`, `editor`, `translator`), und bewirkt bei Angabe des Attributwerts `yes`, dass der Familienname vor Vornamen angegeben wird.
- language** kann bei allen Feldern verwendet werden, um sie in einer anderen Sprache als der des gesamten Eintrags zu setzen. Für `<Attributwert>` kann jede vom Paket `babel` im Dokument angezogene Sprache angegeben werden, um die für das Feld zu verwendenden Trennmuster festzulegen.

13.2.6 Beispiel einer `.ltb`-Datei

Das folgende Beispiel zeigt eine vollständige `.ltb`-Datei mit fünf Einträgen der Typen `article`, `book`, `incollection`, `manual` und `report`. Beim Formatieren der `.ltb`-Datei wird ein Dokument erzeugt, das lediglich aus dem Literaturverzeichnis besteht. Damit dokumentiert die Datei zugleich die in ihr gesammelte Literatur.

```

1  \documentclass{article}
2  \usepackage[german,ngerman]{babel}
3  \usepackage{amsrefs}
4  \pagestyle{empty}
5  \setlength{\textwidth}{11cm}
6  \setlength{\textheight}{17cm}
7  \begin{document}
8  \begin{bibdiv}
9  \begin{biblist}
10 \DefinePublisher{ams}{AMS}{American
11 Mathematical Society}{Providence}
12 \newcommand{\LyX}{L\kern-.1667em%
13 \lower.25em\hbox{Y}\kern-.125emX\@}
14
15 \bib{LaTeXbegleiter}{book}{
16   title={Der \LaTeX-Begleiter},
17   author={Goossens, Michael},
18   author={Mittelbach, Frank},
19   author={Samarin, Alexander},
20   publisher={Addison-Wesley Publishing Company},
21   edition={1.~Auflage},
22   address={Bonn/Paris/Reading, Mass. [u.\,a.]},
23   date={1994},
24   isbn={3-89319-646-3},
25   translator={Kraft, Claudia},
26   translator={Stiels, Rebecca},

```

```

27     note={\LaTeXe{} Standardreferenz},
28     language={Deutsch},
29     hyphenation={german},
30 }
31
32 \bib{AMSJones2004UG}{report}{
33     title={User's Guide to the \texttt{amsrefs} Package},
34     author={Jones, David M.},
35     date={2004-06-30},
36     edition={Version 2.0},
37     organization={American Mathematical Society},
38     address={Providence, RI},
39     note={Benutzerdokumentation zu den AMS-Erweiterungen
40         f\"ur Literaturangaben}*{language={german}},
41     language={Englisch},
42     hyphenation={USenglish},
43 }
44
45 \bib{KMW}{incollection}{
46     author={Kac, V.~G.},
47     author={Moody, R.~V.},
48     author={Wakimoto, M.},
49     title={On  $E_{10}$ },
50     date={1988},
51     booktitle={Differential geometrical methods in
52         theoretical physics (Como, 1987)},
53     publisher={Kluwer Acad. Publ.},
54     address={Dordrecht},
55     pages={109\ndash 128},
56     language={Englisch},
57     hyphenation={USenglish},
58 }
59
60 \bib{LyXIntro}{manual}{
61     title={Einführung in LyX oder Das Einmaleins der
62         LyX-Dokumentation},
63     author={\LyX{} Team},
64     translator={S\"utterlin, Peter},
65     translator={Haase, Hartmut},
66     date={2003},
67     note={Teil der Online-Dokumentation},
68     language={german},
69     hyphenation={ngerman},
70 }
71
72 \bib{Rob}{article}{
73     author={Robinson, D.},
74     title={On representations of the symmetric group},
75     date={1938},

```

```

76     journal={Amer. J. Math.},
77     volume={60},
78     pages={745\ndash 760},
79     language={Englisch},
80     hyphenation={USenglish},
81 }
82 \end{biblist}
83 \end{bibdiv}

```

Literatur

- [1] Michael Goossens, Frank Mittelbach, and Alexander Samarin, *Der L^AT_EX-Begleiter*, 1. Auflage, translated by Claudia Kraft and Rebecca Stiels, Addison-Wesley Publishing Company, Bonn/Paris/Reading, Mass. [u. a.], 1994 (Deutsch). L^AT_EX 2_ε Standardreferenz.
- [2] David M. Jones, *User's Guide to the amsrefs Package*, Version 2.0, American Mathematical Society, Providence, RI, June 30, 2004 (Englisch). Benutzerdokumentation zu den AMS-Erweiterungen für Literaturangaben.
- [3] V. G. Kac, R. V. Moody, and M. Wakimoto, *On E₁₀*, Differential geometrical methods in theoretical physics (Como, 1987), 1988, pp. 109–128 (Englisch).
- [4] L^AX Team, *Einführung in LyX oder Das Einmaleins der LyX-Dokumentation*, translated by Peter Sütterlin and Hartmut Haase, 2003 (german). Teil der Online-Dokumentation.
- [5] D. Robinson, *On representations of the symmetric group*, Amer. J. Math. **60** (1938), 745–760 (Englisch).

[Buch1/AmSLayout/amsref-ltb.tex]

13.2.7 Zitieren

Der `\cite`-Befehl von L^AT_EX setzt einen Verweis auf das Literaturverzeichnis durch Angabe der von L^AT_EX oder dem Anwender vergebenen Zitiermarke in eckigen Klammern. Andere Formen von Verweisen auf die Literatur werden nicht unterstützt.

Das Paket `amsrefs` unterstützt auch andere Verweis-Formen. Für mehrere alternative Formen werden Befehle zur Verfügung gestellt. Für den `\cite`-Befehl wird eine alternative Form bereitgestellt, mit der Zusätze bei Verwendung in optionalen Argumenten einfacher angegeben werden können. Für Mehrfachverweise werden spezielle Befehle angeboten, die verwendet werden sollten, auch wenn ein einzelner `\cite`-Befehl mehrere Literaturverweise enthalten kann. Bei den folgenden Zitierbefehlen steht *<Marke>* für eine Marke im Sinne von BIB_TE_X. *<Zusatz>* ist beliebiger Text wie im optionalen Argument des `\cite`-Befehls. *<Autor>* steht für einen einzelnen Autor oder eine Autorenliste, die gegebenenfalls abgekürzt ist.

```
\cite{<Marke>}[*{<Zusatz>}]
```

Alternative Form von `\cite[<Zusatz>]{Marke}`
amrefs akzeptiert beide Formen.

```
\citelist{\cite{<Marke>}\cite{<Marke>}...}
```

Setzt einen Mehrverweis, der aus mehreren `\cite`-Befehlen aufgebaut wird. Bei Bedarf kann zu einzelnen `\cite`-Befehlen ein nur für dieses Verweis gültiger `<Zusatz>` angegeben werden. Die einzelnen `\cite`-Befehle können durch Leerzeichen getrennt werden; andere Einschübe sind nicht vorgesehen.

```
\cites{<Marke>[, <Marke>]...}
```

Alternative Form von
`\citelist{\cite[<Zusatz>]{Schl}}\cite{<Schl>}...`

```
\nocite{<Marke>[, <Marke>]...}
```

Behält seine Funktion, wirkt aber nur auf *nachfolgende* `\bibselect`-Befehle. Eine feinere Kontrolle erlaubt die Verwendung von `\bibselect*`.

Erweiterungen im Fall der Option `author-year`

In diesem Fall setzt `\cite` die Form `(<Autor>, <Jahr>)`.
Autorenlisten werden abgekürzt.

```
\ycite{<Marke>}[*{<Zusatz>}]
```

Gibt nur das Jahr in der Form `(<Jahr>)` als Verweis an und wird verwendet, wenn die Namen der Autoren als Teil des Satzes genannt sind.

```
\ycites{<Marke>[, <Marke>]...}
```

Mehrfachverweis wie `\cites`, wobei aber nur das Jahr angegeben wird.

Entspricht
`\citelist{\ycite{<Marke>}}\ycite{<Marke>}...`

```
\ocite{<Marke>}[*{<Zusatz>}]
```

Verweist in der Form `<Autor> (<Jahr>)`.
Autorenlisten werden abgekürzt.

```
\ocites{<Marke>[, <Marke>]...}
```

Mehrfachverweis wie `\cites`, wobei aber nur Autoren angegeben werden.

Entspricht
`\citelist{\ocite{<Marke>}}\ocite{<Marke>}...`

```
\citeauthor{<Marke>}[*{<Zusatz>}]
```

Verweist durch Angabe der Autorenlist in der Form *<Autor>*.

```
\fullcite{<Marke>}[*{<Zusatz>}]
```

Wie `\cite`, aber sämtliche Autoren werden aufgeführt.

```
\fullocite{<Marke>}[*{<Zusatz>}]
```

Wie `\ocite`, aber sämtliche Autoren werden aufgeführt.

Das folgende Beispiel zeigt die Formen, in denen die Befehle bei Angabe der Option `author-year` auf die Literatur verweisen. Das Literaturverzeichnis wurde aus dem vorigen Beispiel übernommen, wobei die Marke `LaTeXbegleiter` zu `Begleiter` abgekürzt wurde.

Hinweis: Zu beachten ist bei diesem Beispiel ausnahmsweise die Reihenfolge der `\usepackage`-Befehle. Lädt man das Paket `babel` nach `amsrefs`, so kommt es zu Fehlermeldungen. Alternativ kann man das Paket `ngerman` bzw. `german` verwenden, dann ist die Reihenfolge wieder egal.

```

9  \cite{Begleiter} \\  

10 \cite{Begleiter}*{Standardreferenz}  

11  

12 Goossens, Mittelbach und Samarin  

13 \ycite{Begleiter} \\  

14 \ycite{Begleiter}*{Standardreferenz}  

15  

16 \ocite{Begleiter} \\  

17 \ocite{Begleiter}*{Standardreferenz}  

18  

19 \citeauthor{Begleiter} \\  

20 \citeauthor{Begleiter}*{Standardreferenz}  

21  

22 \citelist{\cite{Begleiter}  

23           \cite{LyXIntro}*{Einführung}  

24           \cite{Rob}}  

25  

26 \cites{Begleiter,LyXIntro,Rob}  

27  

28 \ycites{Begleiter,LyXIntro,Rob}  

29  

30 \ocites{Begleiter,LyXIntro,Rob}  

31  

32 \fullcite{Begleiter} \\  

33 \fullcite{Begleiter}*{Standardreferenz}  

34  

35 \fullocite{Begleiter} \\  

36 \fullocite{Begleiter}*{Standardreferenz}

```

(Goossens et al., 1994)
 (Goossens et al., 1994, Standardreferenz)
 Goossens, Mittelbach und Samarin (1994)
 (1994, Standardreferenz)
 Goossens et al. (1994)
 Goossens et al. (1994, Standardreferenz)
 Goossens et al.
 Goossens et al., Standardreferenz
 (Goossens et al., 1994; L^AT_EX Team, 2003, Einführung; Robinson, 1938)
 (Goossens et al., 1994; L^AT_EX Team, 2003; Robinson, 1938)
 (1994, 2003, 1938)
 Goossens et al. (1994), L^AT_EX Team (2003), and Robinson (1938)
 (Goossens, Mittelbach, and Samarin, 1994)
 (Goossens, Mittelbach, and Samarin, 1994)*Standardreferenz
 Goossens, Mittelbach, and Samarin (1994)
 Goossens, Mittelbach, and Samarin (1994)*Standardreferenz

Literatur

Goossens, Michael, Frank Mittelbach, and Alexander Samarin. 1994. *Der L^AT_EX-Begleiter*, 1. Auflage, translated by Kraft, Claudia and Rebecca Stiels, Addison-Wesley Publishing Company, Bonn/Paris/Reading, Mass. [u. a.] (Deutsch). L^AT_EX 2 ϵ Standardreferenz.

Jones, David M. 2004. *User's Guide to the amsrefs Package*, Version 2.0, American Mathematical Society, Providence, RI (Englisch). Benutzerdokumentation zu den AMS-Erweiterungen für Literaturangaben.

Kac, V. G., R. V. Moody, and M. Wakimoto. 1988. *On E₁₀*, Differential geometrical methods in theoretical physics (Como, 1987), pp. 109–128 (Englisch).

L^AT_EX Team. 2003. *Einführung in LyX oder Das Einmaleins der LyX-Dokumentation*, translated by Sütterlin, Peter and Hartmut Haase (german). Teil der Online-Dokumentation.

Robinson, D. 1938. *On representations of the symmetric group*, Amer. J. Math. **60**, 745–760 (Englisch).

13.2.8 Anpassung von Bibliografestilen

Die äußere Form eines Literaturverzeichnisses wird vor allem durch die Darstellung der unterschiedlichen Eintragsarten und die dabei berücksichtigten Informationen bestimmt. Anpassen eines Bibliografestils bedeutet also vor allem, geeignete Eintragsarten und ihre Darstellung zur Verfügung zu stellen. Da `amsrefs` in Form eines $\text{\LaTeX}$ -Pakets implementiert ist, sollten Anpassungen als eigenes Paket realisiert werden, das auf `amsrefs` aufbaut.

Zur Definition der Eintragsarten und ihrer Umsetzung verwendet `amsrefs` einen eigenen Befehl, der auch dem Benutzer zur Verfügung steht:

```
\BibSpec{<Eintragstyp>}{%
  [+{<Trennung>}{<Text>}{<Feld>}]
  ...
}
```

Das erste Argument legt den Eintragstyp fest, dessen Definition geändert oder der neu festgelegt wird. Das zweite Argument des `\BibSpec`-Befehls legt fest, welche Felder berücksichtigt werden und wie sie zu formatieren sind.

Für jedes zu berücksichtigende Feld ist dabei eine Zeile wie in der Definition anzugeben: Die Angaben beginnen mit dem Zeichen `+`, dem drei Argumente folgen. Das Argument `<Trennung>` wird nur dann verwendet, wenn das angegebene `<Feld>` Angaben enthält. Es dient zur Angabe von Trennzeichen, die die Ausgaben dieses Felds vom vorhergehenden Feld trennen. Ist `<Feld>` leer, entfällt auch `<Trennung>`.

`<Text>` enthält Abstände und Anderes, das zwischen `<Trennung>` und dem Inhalt des Felds gesetzt werden soll. `<Text>` kann mit einem Befehl enden, dem dann der Inhalt des Felds als (letztes) Argument übergeben wird. Neben Befehlen wie `\textit` zur Schriftumschaltung werden häufig die im Folgenden beschriebenen Befehle `\Print...`, `\parenthesize` oder `\bibquotes` verwendet.

Schließlich ist das Argument `<Feld>` der Name des Felds, dessen Formatierung in der Zeile beschrieben wird.

Felder, zu denen diese Zeile nicht angegeben ist, werden ignoriert. Neue Felder müssen vor Verwendung im `\BibSpec`-Befehl vereinbart werden.

Definierten Eintragstypen können weitere Typnamen zugeordnet werden. `amsrefs` definiert beispielsweise die Typen `periodical`, `collection`,

proceedings, manual, miscellaneous und unpublished als Synonyme für book und techreport als Synonym für report. Der Befehl zur Definition eines Synonyms kann auch bei eigenen Anpassungen verwendet werden:

```
\BibSpecAlias{<Neuer Typ>}{<Eintragtyp>}
```

definiert Neuer Typ als Synonym für einen bereits (mit \BibSpec oder \BibSpecAlias) definierten Eintragtyp.

Es ist allerdings keine Möglichkeit vorgesehen, nachträglich kleinere Änderungen an einem Eintragtyp vorzunehmen und die anderen Teile unverändert zu übernehmen. Das ist nur im Rahmen einer kompletten Neudefinition des Eintragsyps möglich. Daher kann \BibSpecAlias nicht zum Vereinbaren eines ähnlichen Eintragsyps verwendet werden.

Bei der Vereinbarung neuer Felder werden vier Varianten unterschieden. Zwei dieser Varianten dienen zum Vereinbaren einfacher und wiederholbarer Felder. Zusammengesetzte Felder sind je nach ihrem Gebrauch als einfaches oder zusammengesetztes Feld zu vereinbaren; die Behandlung der untergeordneten Felder ist durch einen im Argument <Text> angegebenen Befehl anzustoßen. Die Befehle haben die folgende Form:

```
\DefineSimpleKey{<Gruppe>}{<Feld>}
\DefineAdditiveKey{<Gruppe>}{<Feld>}{<Wrapper>}
\DefineSupersedingKey{<Gruppe>}{<Feld>}
\DefineDummyKey{<Gruppe>}{<Feld>}
```

Die ersten beiden Befehle vereinbaren ein einfaches und ein wiederholbares Feld. Der Befehl \DefineSupersedingKey vereinbart ein Feld, das zwar mehrfach angegeben werden kann, bei dem aber nur die letzte Angabe übernommen wird; keines der in amsrefs definierten Felder ist von diesem Typ.

Der letzte Befehl vereinbart ein Feld, das vom \BibSpec-Befehl als nicht leer betrachtet und dessen Formatierung daher stets übernommen wird. Ein eventuell vorhandener Inhalt eines solchen Felds wird jedoch ignoriert.

Alle vier Befehle haben die gleichen ersten beiden Argumente. Das erste Argument legt eine Gruppenzugehörigkeit fest. amsrefs verwendet die beiden Gruppen bib für Felder und prop für die in 13.2.5 beschriebenen Feldeigenschaften, die mit den gleichen Befehlen vereinbart werden. Bei Feldern ist stets bib anzugeben. Das zweite Argument ist der Name des neuen Felds oder – im Fall <Gruppe>=prop – der neuen Feldeigenschaft.

Das Argument *<Wrapper>* beim Befehl `\DefineAdditiveKey` ist ein Befehl, dem als Argument den Inhalt eines einzelnen Felds *<Feld>* mitgegeben wird.

Die folgenden Befehle haben jeweils ein einzelnes Argument, in der Regel den Namen eines Felds, der wie der Befehl selbst als Argument im `\BibSpec`-Befehl angegeben ist.

`\bibquotes`

Setzt das angegebene Argument in Anführungszeichen. Folgt ein Komma oder ein anderes Satzzeichen, werden die schließenden Anführungszeichen erst nach dem Satzzeichen gesetzt. Durch Angabe der Option `logical-quotes` kann dieses Verhalten so geändert werden, dass die schließenden Anführungszeichen in jedem Fall dem Argument direkt folgen.

`\parenthesize`

Setzt das angegebene Argument in runde Klammern. Dieser Befehl wird benötigt, da es beim `\BibSpec`-Befehl nicht vorgesehen ist, Material anzugeben, das dem Inhalt des Felds folgt.

`\PrintAuthors`

Formatiert und setzt die Liste der Autoren oder – bei gleichen Autoren wie im vorhergehenden Eintrag – den Inhalt des Befehls `\bysame`.

`\PrintBook`

Setzt die Angaben des zusammengesetzten Felds `book`. Die Form der Ausgabe wird durch den internen Eintragstyp `innerbook` festgelegt.

`\PrintConference`

Setzt die Angaben des zusammengesetzten Felds `conference`. Der Titel der Konferenz wird direkt gesetzt. Die übrigen Angaben zur Konferenz setzt der Befehl `\PrintConferenceDetails`.

`\PrintConferenceDetails`

Setzt alle Angaben zu einer Konferenz mit Ausnahme des Titels.

`\PrintContributions`

Setzt die Angaben in den zusammengesetzten Feldern `contribution`.

`\PrintDate`

Konvertiert ein nach ISO 8601 angegebenes Datum in die Form, die für den aktuellen Eintrag vorgesehen ist, und setzt diese Datumsangabe in runden Klammern.

\PrintDateB

Konvertiert ein nach ISO 8601 angegebenes Datum wie `\PrintDate`, setzt die Datumsangabe aber ohne Klammern.

\PrintDatePosted

Konvertiert wie `\PrintDate` ein nach ISO 8601 angegebenes Datum und setzt die Datumsangabe in der Form `„..., posted on <Datum>“`.

\PrintDatePV

Ausgabe eines Datums mit Hilfe von `\PrintDate` oder `\PrintDatePosted` in Abhängigkeit von Angaben in den Feldern `doi` und `volume`. Fehlen Angaben im Feld `doi`, wird das Datum mit `\PrintDate` ausgegeben. Andernfalls wird geprüft, ob das Feld `volume` leer ist, und in diesem Fall wird das Datum mit `\PrintDatePosted` gesetzt; andernfalls wird `\PrintDate` zur Ausgabe verwendet.

\PrintDOI

Setzt die Angaben im Feld `doi`, denen `„DOI“` vorangestellt wird. Enthält auch das Feld `volume` Angaben, wird `„..., (to appear in print)“` hinzugefügt.

\PrintEdition

Konvertiert eine im Feld `edition` angegebene Auflagennummer mit Hilfe des Befehls `\CardinalNumeric` in eine Kardinalzahl (etwa 2 in `2nd`), hängt den im Befehl `\editiontext` hinterlegten Text an (Voreinstellung: `ed.`) und setzt das Ergebnis. Andernfalls wird der Inhalt des Felds `edition` gesetzt.

\PrintEditorsA

Formatiert und setzt die Liste der Herausgeber. Je nach Anzahl der Herausgeber wird `(ed.)` oder `(eds.)` an die Namensliste angehängt.

\PrintEditorsB

Eine Variante des `\PrintEditorsA`-Befehls, die je nach Anzahl der Herausgeber aber `, ed.` oder `, eds.` anhängt und die gesamte Liste in runde Klammern setzt. Diese Variante wird beispielsweise beim Typ `article` verwendet, um die Herausgeber im Feld `proceedings` oder `collection` zu setzen.

\PrintEditorsC

Auch dies ist eine Variante von `\PrintEditorsA`. Hier wird die Liste der Herausgeber von `Edited by` eingeleitet. Diese Form wird verwendet, wenn die Beiträge der Herausgeber als untergeordnet betrachtet werden und die Veröffentlichung nicht unter dem Namen der Herausgeber erfolgte.

\PrintISBNs

Setzt die Angaben in den Feldern `isbn`, denen der Text `ISBN` vorangestellt wird.

\PrintPartials

Setzt die Angaben in den Feldern `partial`.

\PrintPrimary

Dieser Befehl analysiert die Angaben zur Veröffentlichung und versucht festzustellen, unter welchem Namen die Veröffentlichung erfolgte. Berücksichtigt werden die Angaben zu Autoren, Herausgebern und Übersetzern. Herausgeber werden nur berücksichtigt, wenn keine Autoren angegeben wurden, und Übersetzer nur dann, wenn weder Autoren noch Herausgeber angegeben wurden. Ist die Suche erfolglos, wird eine Warnung ausgegeben. Die Liste der gefundenen Namen wird von `\sameauthors` daraufhin überprüft, ob der vorhergehende Eintrag im Literaturverzeichnis unter den gleichen Namen veröffentlicht wurde, und im Fall der Übereinstimmung werden die Namen durch den Inhalt von `\bysame` ersetzt.

Dieser Befehl ist gedacht für Eintragstypen, bei denen Autoren fehlen und statt dessen Herausgeber oder Übersetzer angegeben werden können.

\PrintReprint

Setzt die Angaben im Feld `reprint`. Sie werden mit `,reprinted in'` eingeleitet.

\PrintReviews

Setzt die Angaben in den Feldern `review`.

\PrintThesisType

Setzt die Typisierungsangaben im Feld `type` eines Eintrags und ist für den Eintragstyp `thesis` gedacht. Zur Behandlung des Felds im zusammengesetzten Feld `contribution` ist der Befehl nicht geeignet.

\PrintTranslation

Setzt die Angaben des zusammengesetzten Felds `translation`.

\PrintTranslatorsA

Formatiert und setzt die Liste der Übersetzer. Der Liste der Namen wird die Angabe `(trans.)` angehängt.

\PrintTranslatorsB

Eine Variante von `\PrintTranslatorsB`, die je nach Anzahl der

Übersetzer , tran. oder , trans. anhängt und die gesamte Liste in runde Klammern setzt. Diese Form wird aktuell nicht verwendet und wird lediglich angeboten, um für Herausgeber und Übersetzer die gleichen Befehle zur Verfügung zu stellen.

\PrintTranslatorsC

Die dritte Variante zur Angabe der Übersetzer, die der Liste der Übersetzer ,translated by‘ voranstellt.

\PrintYear

Setzt das Jahr eines nach ISO 8601 angegebenen Datums in den Text, wie es in den Standardstilen für den Eintragstyp `book` vorgesehen ist.

Die Befehle dieser Gruppe legen Bezeichnungen oder spezielle Formen für die Ausgabe fest.

\bysame

Definiert eine horizontale Linie der Länge 3em, die der Befehl `\sameauthors` statt der Autorenliste setzt, wenn der vorhergehende Eintrag im Literaturverzeichnis genau die gleichen Autoren hat wie der aktuelle.

\editiontext

Text, der einer Auflagennummer (Feld `edition`) folgt (voreingestellt: `ed.`).

\etalchar

Das Satzzeichen nach `etaltext`.

\etaltext

Voreingestellt auf `et al.`

\issuetext

Text, der einer Zeitschriftennummer (Feld `volume`) vorangestellt wird (voreingestellt: `no.`).

\SubEtal

expandiert zu `\etaltext` ohne Berücksichtigung des Arguments.

\voltext

Text, der einer Bandnummer (Feld `volume`) vorangestellt wird (voreingestellt: `vol.` oder `Vol.` je nachdem, ob das Feld `series` leer ist oder nicht (das Feld `series` wird mit einem Punkt abgeschlossen).

Die folgenden Befehle sind für die Aufbereitung der Feldinhalte gedacht und werden bei der Definition der Befehle zum Setzen der Felder verwendet.

\BibField{<Feld>}

Das angegebene Argument muss der Name eines Felds sein. Der Befehl liefert den Inhalt des angegebenen Felds.

\CardinalNumeric

Liefert die (englische) Kardinalzahl zu einer angegebenen Zahl, also 1st für 1, 2nd für 2, 3rd für 3 und n th für $n \geq 4$.

\DashPages{<Seiten>}

Dieser Befehl untersucht, ob es sich bei dem angegebenen Argument *<Seiten>* – bestehend aus einer oder mehreren Seiten oder einem Seitenbereich – um eine einzelne Seite oder um mehrere Seiten handelt. Scheint es sich um eine einzelne Seite zu handeln, wird der Text *p.* gefolgt vom Argument gesetzt. Andernfalls wird dem angegebenen Argument der Text *pp.* vorangestellt.

\IfEmptyBibField{<Feld>}{<Feld-leer>}

{<Feld-nichtleer>}

<Feld> muss der Name eines Felds sein. Der Befehl prüft, ob das angegebene Feld leer ist, und liefert in diesem Fall *<Feld-leer>*. Andernfalls liefert der Befehl *<Feld-nichtleer>*.

\nopunct

Dieser Befehl bewirkt, dass ein folgendes Satzzeichen entfällt, wenn es mit dem internen Befehl `\@addpunct` hinzugefügt wurde.

\Plural{<Argument>}

Setzt *<Argument>*, falls für das Feld, bei dem der Befehl verwendet wird, mehrere Angaben vorliegen, etwa mehrere Autoren in Feldern *author* oder mehrere Herausgeber in Feldern *editor*.

\PrintSeries

Ein ziemlich komplexer Befehl, der verwendet wird, um eine Liste gleichartiger Angaben zu setzen.

\SingularPlural{<Singular>}{<Plural>}

Prüft, ob für das gerade bearbeitete Feld mehrer Angaben vorliegen, und setzt in diesem Fall *<Plural>*. Andernfalls wird *<Singular>* gesetzt.

\SwapBreak{<Argument>}

Entfernt allen Leerraum direkt vor dem Befehl und fügt an der aktuellen Stelle *<Argument>* ein, das mit einem Satzzeichen beginnen sollte. Befehle zum Zeilenumbruch wie `\linebreak` am Ende des vorhergehenden Texts werden hinter das angegebene Argument verschoben.

`\sameauthors`

Legt das Verhalten fest, wenn der vorhergehende Eintrag im Literaturverzeichnis genau die gleichen Autoren hat wie der aktuelle.

`\eprint{<Argument>}`

Befehl zum Setzen eines Verweises $\langle\textit{Argument}\rangle$ auf eine Veröffentlichung im Internet. Aktuell wird `\eprint` zum Setzen des Inhalts des Felds `eprint` verwendet. In der Annahme, dass es sich bei dem Verweis um eine URL handelt, wird der Befehl als Synonym für den Befehl `\url` verwendet.

`\eprintpages{<Seiten>}`

Setzt das angegebene Argument $\langle\textit{Seiten}\rangle$. Enthält `eprint` Angaben und ist `journal` nicht angegeben, wird zusätzlich `pp.` gesetzt, da es sich bei Angaben in `pages` in diesem Fall vermutlich um die Anzahl der Seiten einer Veröffentlichung im Internet handelt.

13.3 Die $\mathcal{AMS}$ -Fonts

In der Mathematik werden viele unterschiedliche Symbole und Fonts verwendet. Beispiele für die Verwendung spezieller Fonts sind etwa die Bezeichnungen der reellen und natürlichen Zahlen oder von Vektor- und Matrixvariablen, die außerdem in mathematischen Veröffentlichungen nicht einheitlich sind.

$\text{\LaTeX}$ stellt eine Reihe von Grundsymbolen und mehrere mathematische Fonts zur Verfügung. Für den Satz komplexer mathematischer Formeln aus den verschiedensten Bereichen sind diese Standardfonts allerdings nicht ausreichend. Für ihre Veröffentlichungen hat die $\mathcal{AMS}$ daher mehrere zusätzliche Fonts entwickelt und zur Verfügung gestellt, die eine Vielzahl zusätzlicher Symbole und auch einige weitere Fonts für den mathematischen Modus enthalten.

Das Paket `amsfonts` ermöglicht den einfachen Zugriff auf die mathematischen Fonts. Für einige (wenige) zusätzliche Symbole werden in diesem Paket auch Befehle definiert, die ihre Verwendung erleichtern. Das Paket `eufrak` stellt ‚Fraktur‘-Buchstaben zur Bezeichnung von Variablen zur Verfügung, falls das Paket `amsfonts` nicht ohnehin benötigt wird. Die Pakete `eucal` und `euscript` ersetzen den standardmäßig vorhandenen kalligrafischen Font im mathematischen Modus durch den Font ‚Euler Script‘, der ebenso wie der Fraktur-Font von Herrmann Zapf im Auftrag der $\mathcal{AMS}$ entwickelt wurde. Der komplette Satz der von der $\mathcal{AMS}$ zur Verfügung gestellten Symbole kann über das Paket `amssymb` genutzt werden.

In einem der beiden Symbolfonts der $\mathcal{AMS}$ finden sich auch die so genannten ‚Blackboard Bold‘ Großbuchstaben. Nach Laden des Pakets `amsfonts` können diese Buchstaben im mathematischen Modus beispielsweise über den Befehl `\Bbb` angesprochen werden.

13.3.1 Paket `amsfonts`

Aufgabe dieses Pakets ist es, die beiden Symbolfonts der $\mathcal{AMS}$ und den Fraktur-Font von Herrmann Zapf zu laden. Es wird auch von anderen Paketen der Sammlung zum Laden der Fonts verwendet. Da das Paket nur für wenige Symbole Befehle definiert, muss man in der Regel das im Folgenden beschriebene Paket `amssymb` laden, wenn man die Symbole verwenden will. Will man hingegen lediglich die in den Fonts enthaltenen ‚Blackboard Bold‘-Buchstaben und die Buchstaben des Fraktur-Fonts nutzen, kann man das Paket mit

```
\usepackage[psamsfonts]{amsfonts}
```

laden. Beim Laden kann die Option `psamsfonts` angegeben werden, um statt der `pk`-Fonts die PostScript-Version der $\mathcal{AMS}$ -Fonts von Inc. Y&Y bzw. Inc. Blue Sky Research zu laden. Andere Optionen sind nicht vorgesehen.

Da die Befehle für die meisten Symbole im Paket `amssymb` vereinbart werden, kann auf die Behandlung der wenigen in diesem Paket vereinbarten Befehle hier verzichtet werden. Daher werden hier nur die Befehle für die ‚Blackboard Bold‘-Buchstaben und die Buchstaben des Fraktur-Fonts beschrieben.

Der ‚Blackboard Bold‘-Font steht im mathematischen Modus zur Verfügung und enthält nur Großbuchstaben. Die Zeichen können im mathematischen Modus mit dem Befehl

```
\mathbb{<Buchstabe(n)>}
```

angesprochen werden. Aus Gründen der Kompatibilität zu früheren Versionen kann man bis auf Weiteres noch den Befehl `\Bbb` mit demselben Argument verwenden. Die Zeichen haben das folgende Aussehen:¹

A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

¹ Es gibt auch andere Versionen dieses Fonts; beispielsweise der im Paket `mathpazo` verwendete PostScript-Font `zplm` entspricht den deutschen Konventionen beim Formelsatz eher (vgl. Beispiel auf Seite 356).

Eine fette Variante des Fonts gibt es nicht.

Im Unterschied dazu stehen die Fraktur-Fonts auch in einer fetten Version zur Verfügung. Sie werden daher in in der Formelversion *bold* (`\mathversion{bold}`) fett gesetzt. Die Fraktur-Buchstaben werden angesprochen durch den folgenden Befehl:

```
\mathfrak{<Zeichen>}
```

Der in früheren Versionen des Pakets verwendete Befehl `\frak{<Zeichen>}` steht aus Kompatibilitätsgründen weiterhin zur Verfügung. Befehle zum Zugriff auf die fetten Fraktur-Buchstaben in der normalen Formelversion enthält dieses Paket nicht. Die Pakete `amsbsy`, `amsmath` und `amstex` definieren jedoch den Befehl `\boldsymbol{. . .}`, der den Zugriff auf die fetten Buchstaben ermöglicht. Die folgende Tabelle gibt eine Übersicht über die Ziffern und Buchstaben. Der vollständige Font ist zusammen mit der fetten Varianten in Abb. 13.2 wiedergegeben.

0	1	2	3	4	5	6	7	8	9					
ℒ	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔
℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔	℔
a	b	c	d	e	f	g	h	i	j	k	l	m		
n	o	p	q	r	s	t	u	v	w	x	y	z		

Um kompatibel mit Dokumenten zu sein, die mit alten Versionen der $\mathcal{AMS}$ -Pakete entwickelt wurden, steht als Synonym für `\mathbf` im mathematischen Modus der alte Befehl `\bold` weiterhin zur Verfügung.

Bei Verwendung der alten Befehle `\Bbb`, `\frak` und `\bold` wird eine Warnung ausgegeben zusammen mit einem Hinweis, welcher aktuelle Befehl verwendet werden sollte.

13.3.2 Paket `amssymb`

In diesem Paket definiert die $\mathcal{AMS}$ eine große Anzahl von Befehlen, um die mathematischen Symbole in den Symbolfonts zu setzen. In den Klassen der $\mathcal{AMS}$ wird das Paket automatisch geladen. Bei Verwendung anderer Dokumentklassen kann es mit

```
\usepackage[psamsfonts]{amssymb}
```

0		1		2		3		4		5		6		7	
8		9		10		11		12		13		14		15	
16		17		18		19		20		21		22		23	
24		25		26		27		28		29		30		31	
32		33		34		35		36		37		38		39	
40		41		42		43		44		45		46		47	
48		49		50		51		52		53		54		55	
56		57		58		59		60		61		62		63	
64		65		66		67		68		69		70		71	
72		73		74		75		76		77		78		79	
80		81		82		83		84		85		86		87	
88		89		90		91		92		93		94		95	
96		97		98		99		100		101		102		103	
104		105		106		107		108		109		110		111	
112		113		114		115		116		117		118		119	
120		121		122		123		124		125		126		127	

(msam)

0		1		2		3		4		5		6		7	
8		9		10		11		12		13		14		15	
16		17		18		19		20		21		22		23	
24		25		26		27		28		29		30		31	
32		33		34		35		36		37		38		39	
40		41		42		43		44		45		46		47	
48		49		50		51		52		53		54		55	
56		57		58		59		60		61		62		63	
64		65		66		67		68		69		70		71	
72		73		74		75		76		77		78		79	
80		81		82		83		84		85		86		87	
88		89		90		91		92		93		94			
96		97										102		103	
104		105		106		107		108		109		110		111	
112		113		114		115		116		117		118		119	
120		121		122		123		124		125		126		127	

(msbm)

Abbildung 13.1: Symbolfonts der $\mathcal{AMS}$

geladen werden, wenn nur die Symbole verwendet werden. Die Option `psamsfonts` kann angegeben werden, um die PostScript-Version der Fonts mit Hilfe des Pakets `amsfonts` zu laden. Die Option wird an `amsfonts` durchgereicht, das von `amssymb` geladen wird, falls es nicht bereits geladen ist. Eigene Optionen sind für das Paket nicht vorgesehen.

Abbildung 13.1 gibt eine Übersicht über die beiden Symbolfonts der $\mathcal{AMS}$. Aus den Zeichen auf den Positionen 57, 75 und 76 im Font `,msam'` werden die Befehle `\dashrightarrow` und `\dashleftarrow` gebildet. Befehle, um sie direkt nutzen zu können, sind nicht vorgesehen. Die Zeichen mit den Nummern 91 und 92 im Font `,msbm'` werden vom Befehl `\widehat` je nach Bedarf verwendet, die mit den Nummern 93 und 94 von `\widetilde`. Andere Befehle stehen für sie nicht zur Verfügung.

Eine fette Version der Fonts enthält die Distribution nicht. Gegebenenfalls können ‚fette Symbole‘ mit Hilfe des Befehls `\pmb` aus dem Paket `ambsy` (siehe 12.2.2 auf Seite 421) emuliert werden.

Die Tabellen 13.2–13.7 auf den folgenden Seiten geben eine Übersicht über die Zeichen in den beiden Fonts und die Befehle, mit denen sie gesetzt werden. Bei den Symbolen in den Tabellen 13.2–13.4 handelt es sich um Relationen. Tabelle 13.5 enthält die zusätzlichen (binären) Operatoren, und Tabelle 13.6 enthält Klammersymbole. Die letzte Tabelle enthält Zeichen, die im mathematischen Modus von Nutzen sein können.

Tabelle 13.2: Pfeile und negierte Pfeile im Paket `amssymb` (Relationen)

$\dashrightarrow$	<code>\dashrightarrow</code>	$\dashleftarrow$	<code>\dashleftarrow</code>
$\Rightarrow$	<code>\Rightarrow</code>	$\Leftarrow$	<code>\Leftarrow</code>
$\Downarrow$	<code>\Downarrow</code>	$\Uparrow$	<code>\Uparrow</code>
$\Leftrightarrow$	<code>\Leftrightarrow</code>	$\Rrightarrow$	<code>\Rrightarrow</code>
$\Rrightarrow$	<code>\Rrightarrow</code>	$\Lleftarrow$	<code>\Lleftarrow</code>
$\twoheadrightarrow$	<code>\twoheadrightarrow</code>	$\twoheadleftarrow$	<code>\twoheadleftarrow</code>
$\rightarrowtail$	<code>\rightarrowtail</code>	$\leftarrowtail$	<code>\leftarrowtail</code>
$\looparrowright$	<code>\looparrowright</code>	$\looparrowleft$	<code>\looparrowleft</code>
$\rightleftharpoons$	<code>\rightleftharpoons</code>	$\leftrightharpoons$	<code>\leftrightharpoons</code>
$\curvearrowright$	<code>\curvearrowright</code>	$\curvearrowleft$	<code>\curvearrowleft</code>
$\circlearrowright$	<code>\circlearrowright</code>	$\circlearrowleft$	<code>\circlearrowleft</code>
$\Rsh$	<code>\Rsh</code>	$\Lsh$	<code>\Lsh</code>
$\upharpoonright$	<code>\upharpoonright</code>	$\downharpoonright$	<code>\downharpoonright</code>
$\upharpoonleft$	<code>\upharpoonleft</code>	$\downharpoonleft$	<code>\downharpoonleft</code>
$\rightsquigarrow$	<code>\rightsquigarrow</code>	$\leftrightsquigarrow$	<code>\leftrightsquigarrow</code>
$\nrightarrow$	<code>\nrightarrow</code>	$\nleftarrow$	<code>\nleftarrow</code>
$\nRrightarrow$	<code>\nRrightarrow</code>	$\nLeftarrow$	<code>\nLeftarrow</code>
$\nleftarrow$	<code>\nleftarrow</code>	$\nLleftarrow$	<code>\nLleftarrow</code>

Tabelle 13.3: Binäre Relationen im Paket amssymb

$\leqslant$	<code>\leqq</code>	$\geqslant$	<code>\geqq</code>
$\leqslant$	<code>\leqslant</code>	$\geqslant$	<code>\geqslant</code>
$\leqslant$	<code>\leqslantless</code>	$\geqslant$	<code>\geqslantgtr</code>
$\approx$	<code>\lesssim</code>	$\approx$	<code>\gtrsim</code>
$\approx$	<code>\lessapprox</code>	$\approx$	<code>\gtrapprox</code>
$\ll$	<code>\lll</code>	$\gg$	<code>\ggg</code>
$\ll$	<code>\llless</code>	$\gg$	<code>\gggtr</code>
$\preccurlyeq$	<code>\preccurlyeq</code>	$\succcurlyeq$	<code>\succcurlyeq</code>
$\curlyeqprec$	<code>\curlyeqprec</code>	$\curlyeqsucc$	<code>\curlyeqsucc</code>
$\prec$	<code>\precsim</code>	$\succ$	<code>\succsim</code>
$\approx$	<code>\precapprox</code>	$\approx$	<code>\succapprox</code>
$\triangleleft$	<code>\vartriangleleft</code>	$\triangleright$	<code>\vartriangleright</code>
$\triangleleft$	<code>\triangleleftteq</code>	$\triangleright$	<code>\trianglerightteq</code>
$\blacktriangleleft$	<code>\blacktriangleleft</code>	$\blacktriangleright$	<code>\blacktriangleright</code>
$\subset$	<code>\subteq</code>	$\supset$	<code>\supseteq</code>
$\subseteq$	<code>\Subset</code>	$\supseteq$	<code>\Supset</code>
$\sqsubset$	<code>\sqsubset</code>	$\sqsupset$	<code>\sqsupset</code>
$\lessgtr$	<code>\lessgtr</code>	$\gtrless$	<code>\gtrless</code>
$\lesseqgtr$	<code>\lesseqgtr</code>	$\gtreqless$	<code>\gtreqless</code>
$\lesseqqgtr$	<code>\lesseqqgtr</code>	$\gtreqqless$	<code>\gtreqqless</code>
$\therefore$	<code>\therefore</code>	$\because$	<code>\because</code>
$\fallingdotseq$	<code>\fallingdotseq</code>	$\risingdotseq$	<code>\risingdotseq</code>
$\smile$	<code>\smallsmile</code>	$\frown$	<code>\smallfrown</code>
$\Vdash$	<code>\Vdash</code>	$\Vdash$	<code>\Vdash</code>
$\vdash$	<code>\vdash</code>	$\multimap$	<code>\multimap</code>
$\circ$	<code>\circeq</code>	$\circ$	<code>\eqcirc</code>
$\dot{=}$	<code>\doteqdot</code>	$\triangleq$	<code>\triangleq</code>
$\doteq$	<code>\Doteq</code>		
$\trianglelefteq$	<code>\trianglelefteq</code>		
$\bumpeq$	<code>\bumpeq</code>	$\Bumpeq$	<code>\Bumpeq</code>
$\thicksim$	<code>\thicksim</code>	$\eqsim$	<code>\eqsim</code>
$\backsimeq$	<code>\backsimeq</code>	$\backsimeq$	<code>\backsimeq</code>
$\thickapprox$	<code>\thickapprox</code>	$\approxeq$	<code>\approxeq</code>
$\multimap$	<code>\multimap</code>		
$\between$	<code>\between</code>	$\vartriangle$	<code>\vartriangle</code>
$\varpropto$	<code>\varpropto</code>	$\pitchfork$	<code>\pitchfork</code>
$\backepsilon$	<code>\backepsilon</code>	$\restriction$	<code>\restriction</code>

Tabelle 13.4: Negierte binäre Relationen im Paket amssymb

$\nless$	$\ngtr$
$\nleq$	$\ngeq$
$\nleqslant$	$\ngeqslant$
$\lneq$	$\lgneq$
$\lneqq$	$\lgneqq$
$\lvertneqq$	$\lvertgvertneqq$
$\nleqq$	$\ngeqq$
$\lnsim$	$\lgnsim$
$\lnapprox$	$\lgapprox$
$\nprec$	$\nsucc$
$\npreceq$	$\nsucceq$
$\precneqq$	$\succneqq$
$\precnsim$	$\succnsim$
$\precnapprox$	$\succnapprox$
$\nsim$	$\ncong$
$\varsubsetneq$	$\varsupsetneq$
$\subsetneqq$	$\supsetneqq$
$\subsetneq$	$\supsetneq$
$\varsubsetneqq$	$\varsupsetneqq$
$\subsetneq$	$\supsetneq$
$\subsetseteq$	$\supsetseteq$
$\ntriangleleft$	$\ntriangleright$
$\ntrianglelefteq$	$\ntrianglerighteq$
$\nparallel$	$\nmid$
$\nshortparallel$	$\nshortmid$
$\nvDash$	$\nvDash$
$\nVDash$	$\nVDash$

Tabelle 13.5: Binäre Operatoren im Paket amssymb

$\boxplus$	$\dot{+}$
$\boxminus$	$\smallsetminus$
$\boxdot$	$\centerdot$
$\boxtimes$	$\divideontimes$
$\ltimes$	$\rtimes$
$\leftthreetimes$	$\rightthreetimes$
$\lessdot$	$\gtrdot$
$\veebar$	$\barwedge$
$\intercal$	$\doublebarwedge$
$\curlyvee$	$\curlywedge$
$\Cup$	$\Cap$
$\doublecup$	$\doublecap$
$\circledcirc$	$\circledast$
$\circledddash$	

Tabelle 13.6: Klammersymbole in Paket amssymb

$\lrcorner$	<code>\ulcorner</code>	$\urcorner$	<code>\urcorner</code>
$\llcorner$	<code>\llcorner</code>	$\lrcorner$	<code>\lrcorner</code>

Tabelle 13.7: Mathematische Symbole im Paket amssymb

$\diagup$	<code>\diagup</code>	$\diagdown$	<code>\diagdown</code>
$\square$	<code>\square</code>	$\blacksquare$	<code>\blacksquare</code>
$\diamond$	<code>\lozenge</code>	$\blacklozenge$	<code>\blacklozenge</code>
∇	<code>\triangledown</code>	$\blacktriangledown$	<code>\blacktriangledown</code>
$\sphericalangle$	<code>\sphericalangle</code>	$\blacktriangle$	<code>\blacktriangle</code>
$\angle$	<code>\angle</code>	$\measuredangle$	<code>\measuredangle</code>
$\complement$	<code>\complement</code>	$\nexists$	<code>\nexists</code>
$\backprime$	<code>\backprime</code>	$\varnothing$	<code>\varnothing</code>
$\maltese$	<code>\maltese</code>	$\bigstar$	<code>\bigstar</code>
$\Finv$	<code>\Finv</code>	$\Game$	<code>\Game</code>
$\mho$	<code>\mho</code>	$\eth$	<code>\eth</code>
$\digamma$	<code>\digamma</code>	$\varkappa$	<code>\varkappa</code>
$\hslash$	<code>\hslash</code>	$\hbar$	<code>\hbar</code>
$\beth$	<code>\beth</code>	$\gimel$	<code>\gimel</code>
$\daleth$	<code>\daleth</code>	$\Bbbk$	<code>\Bbbk</code>
$\circledR$	<code>\circledR</code>	$\circledS$	<code>\circledS</code>
$\yen$	<code>\yen</code>	$\checkmark$	<code>\checkmark</code>

13.3.3 ‚Euler Fraktur‘-Font

Die Sammlung amfonts stellt auch die ebenfalls von Herrmann Zapf entwickelten ‚Fraktur‘-Buchstaben und einige Sonderzeichen zur Verfügung. Möchte man nur diese Zeichen setzen, aber nicht das (komplette) Paket amfonts laden, kann man das Paket eufrak laden mit

```
\usepackage[psamfonts]{eufrak}
```

Werden beide Pakete geladen, wird eine Warnung ausgegeben, die darauf aufmerksam macht, dass eufrak redundant ist, wenn amfonts geladen ist. Durch Angabe der Option psamfonts kann man die PostScript-Version der Fonts wie beim Paket amfonts beschrieben laden.

Abbildung 13.2 zeigt die normale und die fette Version der der Fraktur-Fonts. In den Positionen 0–9 des Fonts finden sich Varianten von einigen Buchstaben. Bei Verwendung der Varianten ist zu beachten, dass auf Position 0 in der normalen Version eine zusätzliche Variante des Buchstabens d zu finden ist, die in der fetten Version fehlt. Ebenso fehlt in der fetten Version die Variante des Buchstabens u an Position 7. Umgekehrt enthält die fette Version an den Positionen 8 und 9 Varianten der Buchstaben y und z, die in der normalen Version fehlen.

0	đ	1	ð	2	f	3	f	4	g	5	ŧ	6	t	7	u
				18	‘	19	’								
		33	!									38	&	39	'
40	(	41	)	42	*	43	+	44	,	45	—	46	.	47	/
48	o	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;			61	=			63	?
		65	Œ	66	Ɔ	67	Ɔ	68	Ɔ	69	Ɔ	70	Ɔ	71	Ɔ
72	ſ	73	ſ	74	ſ	75	ſ	76	ſ	77	ſ	78	ſ	79	ſ
80	ſ	81	ſ	82	ſ	83	ſ	84	ſ	85	ſ	86	ſ	87	ſ
88	ſ	89	ſ	90	ſ	91	[			93	]	94	^		
		97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z					125	"			127	l

(eufm)

8	ſ	1	ð	2	f	3	f	4	g	5	ŧ	6	t		
		9	3												
				18	‘	19	’								
		33	!									38	&	39	'
40	(	41	)	42	*	43	+	44	,	45	—	46	.	47	/
48	o	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;			61	=			63	?
		65	Œ	66	Ɔ	67	Ɔ	68	Ɔ	69	Ɔ	70	Ɔ	71	Ɔ
72	ſ	73	ſ	74	ſ	75	ſ	76	ſ	77	ſ	78	ſ	79	ſ
80	ſ	81	ſ	82	ſ	83	ſ	84	ſ	85	ſ	86	ſ	87	ſ
88	ſ	89	ſ	90	ſ	91	[			93	]	94	^		
		97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z					125	"			127	l

(eufb)

Abbildung 13.2: ‚Fraktur‘-Font von Herrmann Zapf

Das Paket definiert den in 13.3.1 auf Seite 485 beschriebenen Befehl `\mathfrak{Teilformel}`, um auf die Zeichen im Font zugreifen zu können. Der dort auch beschriebene Befehl `\frak` wird jedoch nicht definiert.

13.3.4 ‚Euler Script‘-Font

Dieser Font von Herrmann Zapf enthält nur Großbuchstaben und einige Sonderzeichen. Er liegt in normaler Strichstärke und einer fetten Version vor.

Das Paket wird mit dem folgenden Befehl geladen:

```
\usepackage[<Option(en)>]{eucal}
```

Die folgenden Optionen können angegeben werden:

psamsfonts

Lädt die PostScript-Version der Fonts wie bei `amsfonts` beschrieben.

mathcal Lädt ‚Euler Script‘ als kalligrafischen Font im mathematischen Modus. Der Befehl `\mathcal` setzt dann Zeichen aus dem Font ‚Euler Script‘.
(Voreinstellung)

mathscr ‚Euler Script‘ wird als eigenständiger mathematischer Font geladen. Zum Zugriff auf die Zeichen des Fonts wird in diesem Fall der Befehl

```
\mathscr{<Zeichen>}
```

definiert, der die Zeichen aus dem Font ‚Euler Script‘ setzt. Der Befehl `\mathcal` setzt die gewohnten kalligrafischen Zeichen.

Alternativ steht das Paket auch unter dem alten Namen `uescript` zur Verfügung. Beim Laden des Pakets mit

```
\usepackage[<Option(en)>]{euscript}
```

können die gleichen Optionen angegeben werden wie bei `eucal`.

Abbildung 13.3 zeigt die normale und die fette Version des Fonts.

0	—														
24	~														
48															
			58	¬		60	ℜ	61	ℑ						
64	ℵ	65	ℳ	66	ℬ	67	ℭ	68	ℰ	69	ℱ	70	ℱ	71	ℊ
72	ℋ	73	ℐ	74	ℐ	75	ℱ	76	ℒ	77	ℳ	78	ℒ	79	℔
80	ℙ	81	ℚ	82	ℛ	83	ℜ	84	℟			86	ℙ	87	ℙ
88	ℳ	89	ℴ	90	ℵ							94	ℒ	95	ℒ
												102	{	103	}
			106									110	\		
120	℟														

(eusm)

0	—														
24	~														
48															
			58	¬		60	℔	61	ℑ						
64	ℵ	65	ℳ	66	ℬ	67	ℰ	68	ℰ	69	ℰ	70	ℱ	71	ℑ
72	ℋ	73	ℑ	74	ℑ	75	℔	76	ℒ	77	ℳ	78	ℒ	79	ℒ
80	ℙ	81	ℒ	82	℔	83	ℑ	84	ℑ	85	ℒ	86	ℒ	87	ℒ
88	ℒ	89	ℑ	90	ℒ							94	ℒ	95	ℒ
												102	{	103	}
			106									110	ℒ		
120	ℑ														

(eusb)

Abbildung 13.3: ‚Script‘-Font von Herrmann Zapf

13.3.5 ‚Euler‘-Font

Auch dieser Font für den Formelsatz wurde von Herrmann Zapf geschaffen, und zwar in einer normalen (`eurm`) und einer fetten Version (`eurb`). Abbildung 13.4 zeigt die beiden Fonts. Zusätzlich entstand der in Abb. 13.5 wiedergegebene Ergänzungsfont `euex`, der eine Reihe von Pfeilen, Klammern und großen Symbolen enthält.

Die $\mathcal{AMS}$ stellt nur die Fonts zur Verfügung. Ein Paket, um auf die Zeichen des Fonts zugreifen zu können, ist nicht Teil der $\mathcal{AMS}$ -Distribution. Das Paket `euler` von Frank Jensen mit Ergänzungen von Frank Mittelbach ermöglicht die Verwendung der Fonts im mathematischen Modus. Es wird in diesem Band allerdings nicht beschrieben, da es nicht zu den Standardpaketen gehört.

0	Γ	1	Δ	2	Θ	3	Λ	4	Ξ	5	Π	6	Σ	7	Υ
8	Φ	9	Ψ	10	Ω	11	α	12	β	13	γ	14	δ	15	ε
16	ζ	17	η	18	θ	19	ι	20	κ	21	λ	22	μ	23	ν
24	ξ	25	π	26	ρ	27	σ	28	τ	29	υ	30	φ	31	χ
32	ψ	33	ω	34	ε	35	ϑ	36	ω					39	φ
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	.	59	,	60	<	61	/	62	>		
64	∂	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z										
96	ℓ	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	ι	124	Ͽ	125	ϖ			127	

(eurm)

0	Γ	1	Δ	2	Θ	3	Λ	4	Ξ	5	Π	6	Σ	7	Υ
8	Φ	9	Ψ	10	Ω	11	α	12	β	13	γ	14	δ	15	ε
16	ζ	17	η	18	θ	19	ι	20	κ	21	λ	22	μ	23	ν
24	ξ	25	π	26	ρ	27	σ	28	τ	29	υ	30	φ	31	χ
32	ψ	33	ω	34	ε	35	ϑ	36	ω					39	φ
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	.	59	,	60	<	61	/	62	>		
64	∂	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z										
96	ℓ	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	ι	124	Ͽ	125	ϖ			127	

(eurb)

Abbildung 13.4: ‚Euler‘-Font von Herrmann Zapf

13.3.6 Paket cmmib57

Dieses kleine Paket kann geladen werden, um die PostScript-Versionen der ‚cmmib‘- und ‚cmbsy‘-Fonts von Inc. Y&Y bzw. Inc. Blue Sky Research statt der Rasterfonts zu verwenden. Das Paket enthält keine Optionen und wird daher mit

8	{	9	}	10	{	11	}	12	{	13	}	14	{	15	}
24	↖	25	↗	26	↘	27	↙								
32	←	33	→	34	↑	35	↓	36	↔	37	↗	38	↘		
40	⇐	41	⇒	42	⇑	43	⇓	44	⇔	45	↖	46	↗		
		49	∞												
56	(	57	)	58	⌊	59	⌋	60	{	61	}	62	'		
72	ℳ	73	Ⓜ												
80	Σ	81	Π	82	∫										
88	Σ	89	Π	90	∫										
96	Π	97	Π												
								108	↕	109	↕				
				122	↙	123	↘	124	↙	125	↘				

(euex)

Abbildung 13.5: Ergänzungen zum ‚Euler‘-Font

```
\usepackage{cmmib57}
```

geladen. Das Paket vereinbart lediglich die PostScript-Fonts und definiert keine Befehle. Es hat die gleiche Funktion wie die `psamsfonts`-Option bei den anderen Paketen, die Fonts laden, beispielsweise beim Paket `amsfonts`.

Teil III

Zusätze

Benutzeroberflächen

Für die praktische Arbeit mit $\text{T}_{\text{E}}\text{X}$ oder $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ wird außer den $\text{T}_{\text{E}}\text{X}$ -Programmen lediglich ein Texteditor zum Erstellen des Dokuments benötigt. Die Befehle zum Aufruf des Editors und der Programme können etwa in einem Befehlsfenster explizit eingegeben werden. Hilfen bei der Eingabe von $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Dokumenten oder zum Aufruf des Formatierers sind in dieser minimalen Arbeitsumgebung in der Regel nicht zu erwarten.

Einfacher wird die Nutzung von $\text{T}_{\text{E}}\text{X}$, wenn man ein Frontend, auch Oberfläche genannt, verwendet. Dann muss man die Befehle zum Aufruf der Programme und dabei erforderliche Angaben nicht kennen. In vielen Oberflächen steht auch eine Online-Hilfe zu den $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Befehlen zur Verfügung. Häufig können $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Befehle aus Menüs ausgewählt und direkt in den Text eingefügt werden. Solche Oberflächen werden für die meisten Betriebssysteme zu ähnlichen Bedingungen wie $\text{T}_{\text{E}}\text{X}$ selbst zur privaten Nutzung kostenfrei angeboten. Eine freie kommerzielle Nutzung ist in einigen Fällen allerdings ausdrücklich ausgeschlossen; bei kommerzieller Nutzung sind daher unbedingt die Lizenzbedingungen zu prüfen. Neben den frei verfügbaren Oberflächen gibt es auch Shareware-Oberflächen wie WINEDT unter Windows. Als Oberfläche ist es dem frei verfügbaren, beispielsweise $\text{T}_{\text{E}}\text{X}$ nicCenter, aber keineswegs überlegen.

Die Anzahl der verfügbaren Oberflächen ist beträchtlich. Abgesehen von der unterschiedlichen Leistungsfähigkeit unterscheiden sie sich vor allem in der Bedienung und der Art und Weise, wie der eingegebene Text angezeigt wird. Das führt zu folgender grober Einteilung der $\text{T}_{\text{E}}\text{X}/\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Oberflächen:

- Textorientierte Oberflächen

Hierzu zählen alle Oberflächen, die in einer reinen Textumgebung – also ohne Grafikunterstützung – lauffähig sind. Grafische Fähigkeiten werden hier allenfalls für einen Preview formatierter $\text{T}_{\text{E}}\text{X}$ -Doku-

mente benötigt. Die Bedienung dieser Oberflächen erfolgt in der Regel über die Tastatur, auch wenn einzelne Arbeiten mit der Maus durchgeführt werden können.

Der wichtigste – und inzwischen möglicherweise einzige – Vertreter dieses Typs von Oberflächen ist Emacs, der unter Unix verbreitet ist und auch unter Windows und anderen wichtigen Betriebssystemen zur Verfügung steht.

Emacs unterstützt bereits in der Standardform das Erstellen von $\LaTeX$ -Dokumenten und $\text{BIB}\TeX$ -,Datenbanken'. Die Erweiterung AUC- $\TeX$ verbessert die Fähigkeiten des Emacs in diesen Bereichen weiter.

- Grafische Oberflächen

Hier steht die Bedienung mit der Maus im Vordergrund, auch wenn alternativ Tastaturbefehle möglich sind. $\LaTeX$ -Befehle und -Umgebungen können über Icons und Pulldown-Menüs in das Dokument eingefügt werden. Der Aufruf der $\TeX$ -Programme erfolgt in der gleichen Weise.

Interessante grafische Oberflächen unter Unix sind $\TeX$ maker und Kile sowie unter Windows $\TeX$ nicCenter. Auch $\text{X}\TeX$ Shell und Xtem zählt man zu den grafischen Oberflächen.

- WYSIWYG-ähnliche Oberflächen

Bei diesen ebenfalls grafischen Oberflächen steht nicht wie bei den anderen Oberflächen die Eingabe für $\LaTeX$ im Vordergrund, sondern die Anzeige des formatierten Dokuments. Bereits beim Erstellen wird eine formatierte Fassung des geschriebenen Texts angezeigt; die zugehörigen $\LaTeX$ -Befehle werden allenfalls dann angezeigt, wenn eine direkte Umsetzung nicht möglich ist.

Oberflächen, die man als WYSIWYG-ähnlich bezeichnen kann, sind bisher ziemlich selten. Immerhin gibt es mit LyX und $\TeX$ macs zwei Vertreter dieses Typs, auch wenn manche Wünsche (noch) offen bleiben.

14.1 Textorientierte Oberflächen

Sicherlich bevorzugen die meisten Anwender grafische Oberflächen für die Arbeit an $\TeX$ -Dokumenten. Eine Reihe von Benutzern zieht jedoch – besonders unter Unix – Tastatureingaben dem Hantieren mit der Maus vor, zumal das Erstellen von Dokumenten ja größtenteils aus der Eingabe von Text über die Tastatur besteht.

Bei Arbeiten an Rechnern, mit denen man nur über das Internet verbunden ist, steht auch nicht immer eine grafikfähige Verbindung zur Verfügung. Auch in diesen Fällen kann man für Arbeiten mit $\text{T}_{\text{E}}\text{X}$ auf textorientierte Oberflächen zurückgreifen.

Leistungsfähige textorientierte Oberflächen unterstützen heutzutage allerdings auch eine Bedienung mit der Maus und bieten Pulldown-Menüs an, sofern sie in einer grafikfähigen Umgebung eingesetzt werden. In diesen Fällen hat man die Wahl zwischen Tastatureingaben und Einsatz der Maus.

Die wichtigste und leistungsfähigste textorientierte Oberfläche dürfte Emacs sein, eine mit Lisp programmierbare allgemeine Benutzeroberfläche für eine Vielzahl von Aufgaben. Emacs unterstützt $\text{T}_{\text{E}}\text{X}$ bereits von Haus aus. Die $\text{T}_{\text{E}}\text{X}$ nischen Fähigkeiten des Emacs werden allerdings wesentlich erweitert durch das Emacs-Paket $\text{AUC}_{\text{T}}\text{E}_{\text{X}}$.

Da, wie schon erwähnt, Emacs der einzige weiter verbreitete Vertreter der textorientierten Oberflächen ist, werden im Folgenden nur dieser und die Erweiterung $\text{AUC}_{\text{T}}\text{E}_{\text{X}}$ behandelt.

14.1.1 GNU Emacs

Emacs wurde vor langer Zeit von Richard Stallman als programmierbarer Texteditor entwickelt. Grafische Oberflächen für Rechner waren damals noch nicht üblich. Bereits in den Anfangszeiten entstanden erste Anpassungen an PC-Betriebssysteme wie DOS.

Im Laufe der Zeit wurde Emacs erweitert und kann heute vielfältig verwendet werden. Unter anderem als Oberfläche für das Bearbeiten von E-Mails und News, zum Verwalten mit Verzeichnissen und Dateien, als Programmieroberfläche für eine Vielzahl von Programmiersprachen sowie HTML/XML einschließlich Unterstützung von Systemen zur Versionsverwaltung, als Terminkalender oder auch schlicht als Shell. Die verschiedenen Arbeiten wie etwa das Editieren einer einzelnen Datei werden in getrennten *Puffern* („Buffer“) durchgeführt, zwischen denen gewechselt werden kann.

Ursprünglich erfolgte die Bedienung von Emacs ausschließlich über Tastaturbefehle. Mit dem Aufkommen grafischer Benutzeroberflächen wurden zunächst von Joachim Martillo und Robert Krawitz Anpassungen an X11 vorgenommen. Heute stehen auch mit der Maus bedienbare Pulldown-Menüs zur Verfügung. ProgEmacs wurde auch an die grafischen Oberflächen von Windows, OS/2 und Mac OS X (MacIntosh) angepasst.

Im Rahmen der Anpassung an X11 wurde zunächst die Variante XEmacs entwickelt – im Unterschied zum GNU Emacs, der erst später angepasst wurde. Dafür wurde beim XEmacs in Kauf genommen, dass Weiterentwicklungen von GNU Emacs bis heute erst mit Verzögerung integriert werden. Da die beiden Varianten sich heute in ihren grafischen Fähigkeiten praktisch nicht unterscheiden, wird hier nur GNU Emacs beschrieben, Weiterentwicklungen für ihn sind schneller verfügbar sind.

Selbst eine oberflächliche Beschreibung der vielfältigen Möglichkeiten von Emacs würde den Rahmen dieses Buchs sprengen, daher beschränkt sich diese Darstellung auf die Bearbeitung von $\text{T}_{\text{E}}\text{X}$ -Dokumenten mit Hilfe von Emacs. Allgemeine Fähigkeiten und Eigenschaften von Emacs werden nur behandelt, soweit sie dabei eine Rolle spielen. Im folgenden Abschnitt werden dann die Erweiterungen durch das Emacs-Paket AUC $\text{T}_{\text{E}}\text{X}$ behandelt.

14.1.1.1 Konfiguration

Emacs besitzt ein integriertes Konfigurationsmenü, das auch von allen wichtigen Emacs-Paketen (auch von dem im nächsten Abschnitt beschriebenen AUC $\text{T}_{\text{E}}\text{X}$) genutzt wird. Das Konfigurationsmenü – die oberste Ebene lässt sich aufrufen über **Options** → **Customize Emacs** → **Top-level Customization Group** – ist hierarchisch organisiert. Auf der obersten Ebene sind bis auf eine Ausnahme nur Verzweigungen zu verschiedenen Gruppen zu finden, zu denen neben ihrem Namen auch eine Kurzbeschreibung angegeben ist. Die Buttons zum Verzweigen zu den Gruppen und für andere Funktionen können mit der Maus angeklickt werden. Ohne Maus kann der Cursor auf einen Button positioniert und anschließend die -Taste zum Auslösen verwendet werden. Für die Arbeit mit $\text{T}_{\text{E}}\text{X}$ sind die Gruppen **Editing** und **WP** → **TeX** (oder bei Verwendung von AUC $\text{T}_{\text{E}}\text{X}$ **WP** → **AUCTeX**) von besonderer Bedeutung. Der Umgang mit dem Konfigurationsmenü ist im Übrigen selbsterklärend, da die jeweiligen Einträge in den Menüs beschrieben sind.

14.1.1.2 Aufbau des Emacs-Fensters und zugehörige Funktionen

Emacs ist unter anderem ein normaler Texteditor, mit dem Textdateien verschiedener Kodierungen, beispielsweise ASCII, Latin $\langle nn \rangle$ oder UTF8, bearbeitet werden können. In der folgenden Beschreibung wird angenommen, dass Emacs in einer grafikfähigen Umgebung mit einer (für Rechtshänder eingestellten) Maus verwendet wird. Im Vordergrund stehen aber die Befehle. Da die Befehle bei Emacs ähnlich wie bei $\text{T}_{\text{E}}\text{X}$

lange Namen haben können, werden hier in der Regel nur die standardmäßig definierten Tastaturkürzel beschrieben. Eine komplette Liste aller Tastaturbefehle (‘Key Bindings’) lässt sich mit der Eingabe `[Strg]-[h][b]` oder mit der Maus über das Pulldown-Menü **Help** → **Describe** → **List Key Bindings** in einem eigenen Puffer anzeigen. Beschreibung und Syntax einer Emacs-Funktion lassen sich über den Tastaturbefehl `[Strg]-[h][f]` abrufen. Weitere Online-Hilfen, darunter Handbücher und Beschreibungen der Funktionen von Emacs, stehen im Pulldown-Menü **Help** zur Verfügung.

Die in Abbildung 14.1 wiedergegebene Anzeige von Emacs ist in mehrere Bereiche aufgeteilt. In den aktuellen Versionen werden unter den Pulldown-Menüs bei einer grafischen Umgebung einige Icons angezeigt, über die besonders wichtige Funktionen erreichbar sind. Danach folgen ein oder mehrere Anzeigebereiche für Puffer, jeweils bestehend aus dem Bearbeitungsbereich, einer Statuszeile darunter und einem *Slider* links daneben. Der *Slider* erlaubt es, den sichtbaren Bereich des Puffers einzustellen. Da der Cursor stets sichtbar ist, lässt sich der Ausschnitt auch über die Cursortasten oder über die Tasten zum Blättern verändern. Unterhalb der Anzeigebereiche für Puffer befindet sich ein – meist deaktivierter – Minipuffer, der (nur) aktiviert wird, wenn Befehle zusätzliche Eingaben erfordern.

Der Anzeigebereich eines Puffers kann durch `[Strg]-[x][2]` vertikal und horizontal durch `[Strg]-[x][3]` geteilt werden. Abbildung 14.1 zeigt eine solche Situation. Beide so entstandenen Anzeigebereiche zeigen zunächst den Inhalt des geteilten Anzeigebereichs, können aber jeweils beliebige andere Puffer anzeigen. `[Strg]-[x][0]` löscht den Anzeigebereich eines Puffers und macht so eine vorhergehende Teilung rückgängig. Auf den Puffer selbst und seinen Inhalt hat diese Operation keinen Einfluss, außer dass er gegebenenfalls zu diesem Zeitpunkt nicht sichtbar ist und erst wieder bearbeitet werden kann, nachdem er erneut in einem der Anzeigebereiche dargestellt wird. Mit `[Strg]-[x][1]` werden alle Anzeigebereiche in einem Fenster mit Ausnahme des aktuellen gelöscht.

In einer grafikfähigen Umgebung besteht die Möglichkeit, gleichzeitig mehrere Emacs-Fenster (‘Frame’) zu öffnen. Jedes dieser Fenster hat die gleiche Aufteilung wie das ursprüngliche und kann mehrere Puffer anzeigen. Der Wechsel zwischen den Fenstern erfolgt in der für die zu Grunde liegende Grafikoberfläche des Betriebssystems üblichen Art.

Zur Ausführung von Emacs-Befehlen ist der Tastaturbefehl `[Esc]-[x]` vorgesehen. Im Anschluss daran kann der auszuführende Befehl im Minipuffer eingegeben werden. Mit `[Strg]-[q]` kann ein gerade ausgeführter Befehl (das gilt auch für die über Tastaturbefehle ausgelöste Emacs-Befehle) abgebrochen werden.

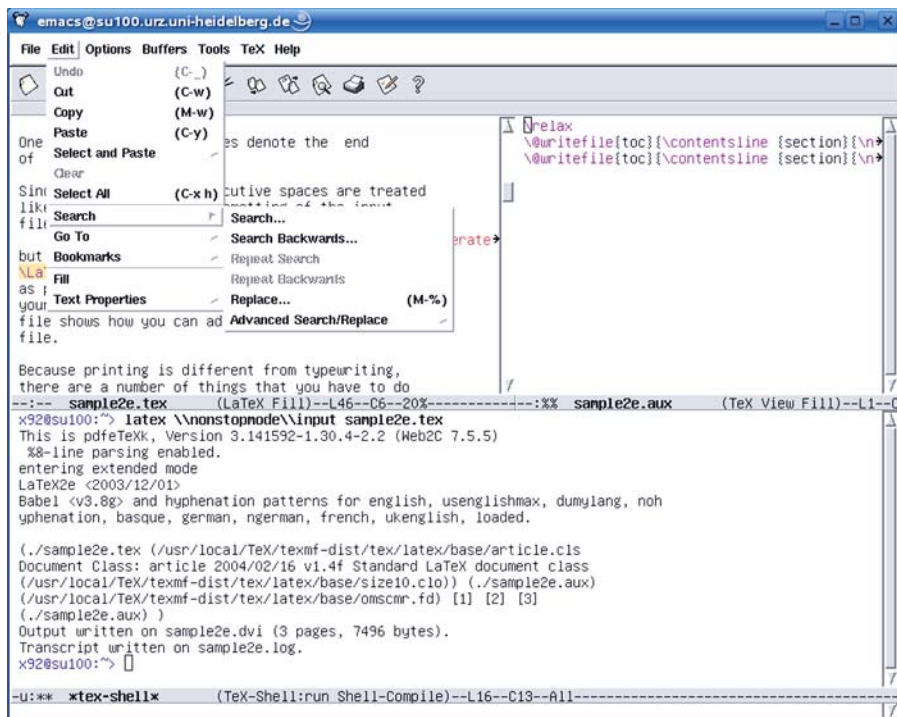


Abbildung 14.1: Emacs beim Editieren mit geöffnetem Pulldown-Menü Edit. Im linken oberen Puffer wird die $\text{\LaTeX}$ -Datei editiert, rechts oben wird die `.log`-Datei angezeigt und unten ist die Ausgabe eines $\text{\LaTeX}$ -Aufrufs zu sehen.

Emacs wird durch `[Strg]-[x]` `[Strg]-[c]` oder durch `File` → `Exit Emacs` beendet. Gibt es – abgesehen vom Puffer `*Scratch*` – Puffer mit Änderungen, die noch nicht gespeichert wurden, wird unter Angabe des Namens des Puffers nachgefragt, ob er gespeichert werden soll. Lehnt man das Speichern mindestens eines dieser Puffer ab, fragt Emacs nochmals nach, ob man wirklich beenden will. Nur wenn diese Frage mit `yes` ausdrücklich bejaht wird, beendet sich das Programm, obwohl sich immer noch ungesicherte Änderungen in (mindestens) einem Puffer befinden.

14.1.1.3 Puffer

Puffer werden von Emacs bei Bedarf automatisch angelegt, etwa um Informationen oder Dateien anzuzeigen, Dateien zu bearbeiten oder für die Ausgaben ausgeführter Befehle oder Programme.

Sie existieren unabhängig davon, ob sie angezeigt werden oder nicht. Zwei Puffer öffnet Emacs standardmäßig: einen für Meldungen (**Messages**), der keine direkten Eingaben durch den Benutzer zulässt, und einen weiteren **Scratch**, der Eingaben erlaubt und für temporäre Editierarbeiten genutzt werden kann. Dem Puffer **Scratch** ist allerdings keine Datei zugeordnet, so dass sein Inhalt beim Verlassen von Emacs nicht gesichert wird, sondern verloren geht.

Allerdings kann dieser Puffer – wie jeder andere – mit `Strg-x Strg-w` oder über `File → Save Buffer As...` in einer Datei gespeichert werden, deren Namen im Eingabepuffer am unteren Rand des Fensters eingegeben werden kann. Zum Speichern des aktuellen (geänderten) Puffers in die zugeordnete Datei – etwa wenn eine Datei zum Bearbeiten geöffnet wurde – kann der Tastaturbefehl `Strg-x Strg-s` oder `File → Save (current buffer)` verwendet werden. Der Tastaturbefehl bietet auch die Möglichkeit, einen anderen Puffer zu speichern. Voreingestellt ist das Speichern des aktuellen Puffers; hier reicht es, die Taste `↵` zu betätigen. Andernfalls kann zuvor noch der Name des zu speichernden Puffers angegeben werden.

Der Tastaturbefehl `Strg-x Strg-b` oder der Menü-Eintrag `Buffers → List All Buffers` öffnet einen speziellen Puffer mit einer Liste aller verfügbaren Puffer. Hier kann ein Puffer durch Anklicken mit der Maus oder über die Cursortasten ausgewählt werden. Nach Drücken der Taste `↵` wird der gewählte Puffer angezeigt. Durch `Strg-x b` gefolgt vom Namen des gewünschten Puffers, Auswahl im Pulldown-Menü `Buffers` oder gleichzeitigem Drücken von `Strg` und der linken Maustaste und Auswahl aus der angezeigten Liste kann auch direkt zum gewünschten Puffer gewechselt werden.

Ein Puffer kann (ohne Sicherung seines Inhalts) beendet werden durch `Strg-x k` oder `File → Close (current buffer)`. Wurden in einem einer Datei zugeordneten Puffer bisher noch nicht gespeicherte Änderungen vorgenommen, wird eine Warnung ausgegeben und eine Bestätigung verlangt. Ohne diese Bestätigung wird der Befehl abgebrochen. Der Anzeigebereich des Puffers wird dabei nicht gelöscht. `Strg-x 4 0` hat die gleiche Wirkung, löscht aber gleichzeitig den Anzeigebereich.

14.1.1.4 Dateien

Beim Öffnen einer Datei zum Bearbeiten mit `Strg-x Strg-f` bzw. `File → Open File...` oder zum Ansehen wird ihr ein Puffer zugeordnet, dem der Name der Datei (ohne Pfadangaben) gegeben wird. Das Öffnen zum Ansehen – ohne Berechtigung zum Schreiben – ist nur über den Verzeichnismodus von Emacs oder den Emacs-Befehl `view-file` möglich,

da standardmäßig ein Tastaturbefehl und ein Eintrag in den Pulldown-Menüs fehlen (mit dem Emacs-Befehl `global-set-key` kann man selbst einen Tastaturbefehl vereinbaren). Der Puffer für die Datei wird im aktuellen Anzeigebereich angezeigt. Mehrere Dateien können gleichzeitig geöffnet sein. Die ihnen zugeordneten Puffer können in beliebigen ‚Frames‘ angezeigt werden, sind also keinem dieser Fenster fest zugeordnet.

Da mehrere Puffer – oder auch ein einzelner Puffer mehrfach – unter bzw. nebeneinander angezeigt werden können, sind – gegebenenfalls abgestimmte – Änderungen in verschiedenen Dateien oder an verschiedenen Stellen in einer Datei möglich.

14.1.1.5 Editierfunktionen

Fließtext wird in Emacs über die Tastatur in der gewohnten Weise eingegeben. Befehle, wie etwa beim Editor vi, werden nicht benötigt, um Text zu schreiben. Dabei hat man die Möglichkeit, zwischen dem Einfüge-Modus, in dem die Eingaben an der Position des Cursors eingefügt werden, und dem Überschreib-Modus zu wählen. Voreingestellt ist der Einfüge-Modus. Der Wechsel zwischen dem Einfüge- und dem Überschreib-Modus erfolgt beim Betätigen der Einfüge-Taste `[Einf]`. Befindet man sich im Überschreib-Modus, wird in der Statuszeile in der runden Klammer neben dem in diesem Puffer verwendeten Emacs-Modus (`(mode)`) zusätzlich der Text `Ovwr` angezeigt.

Das Zeichen an der Cursorposition kann mit `[Entf]` unabhängig vom aktuellen Modus entfernt werden. Der rechts stehende Text rutscht dabei um ein Zeichen nach links. Die Taste `[Backspace]` löscht das Zeichen links vom Cursor und verschiebt im Einfüge-Modus den Text ab der Cursorposition um ein Zeichen nach links; im Überschreib-Modus wird das gelöschte Zeichen durch ein Leerzeichen ersetzt.

Textbereiche können durch Überstreichen mit der linken Maustaste oder durch Bewegen des Cursors mit `[Shift]–[<Cursor>]` markiert werden. Markiert wird dabei fortlaufender Text. Markieren von rechteckigen Blöcken ist mit Hilfe des Emacs-Pakets CUA möglich, das aber üblicherweise nicht Teil der Emacs-Distributionen ist. Die markierten Bereiche bilden die Grundlage für die ‚Cut-and-Paste‘-Operationen.

Die üblichen ‚Cut-and-Paste‘-Operationen stehen in dem in Abbildung 14.1 gezeigten Pulldown-Menü `Edit` und als Tastaturbefehle zur Verfügung: Kopieren über `Edit → Copy` oder `[Esc]–[w]` sowie Ausschneiden (jeweils des markierten Bereichs) über `Edit → Cut` oder `[Strg]–[w]` und Einfügen (des zuletzt kopierten oder ausgeschnittenen Bereichs) über `Edit → Paste` oder

Strg–**y** . Über **Edit** → **Select and Paste** können früher kopierte oder ausgeschnittene Bereiche ausgewählt und eingefügt werden. Das Löschen des markierten Bereichs ist über **Entf** möglich. Der gesamte Puffer lässt sich über **Edit** → **Select All** (**Strg**–**x** **h**) und anschließend **Edit** → **Clear** löschen.

Vorgenommene Änderungen in einem Puffer können über **Edit** → **Undo** oder **Strg**–**z** schrittweise rückgängig gemacht werden. Die Anzahl der Schritte, die Emacs sich merkt, ist vor allem durch den verfügbaren Speicher beschränkt. Der ‚Undo‘-Befehl hat zugleich die Funktion eines ‚Redo‘-Befehls, da ein ‚Undo‘-Befehl auch ein Befehl ist, der selbst rückgängig gemacht werden kann. Der Wechsel zwischen ‚Undo‘ und ‚Redo‘ erfolgt dabei durch eine Eingabe oder eine Cursorbewegung.

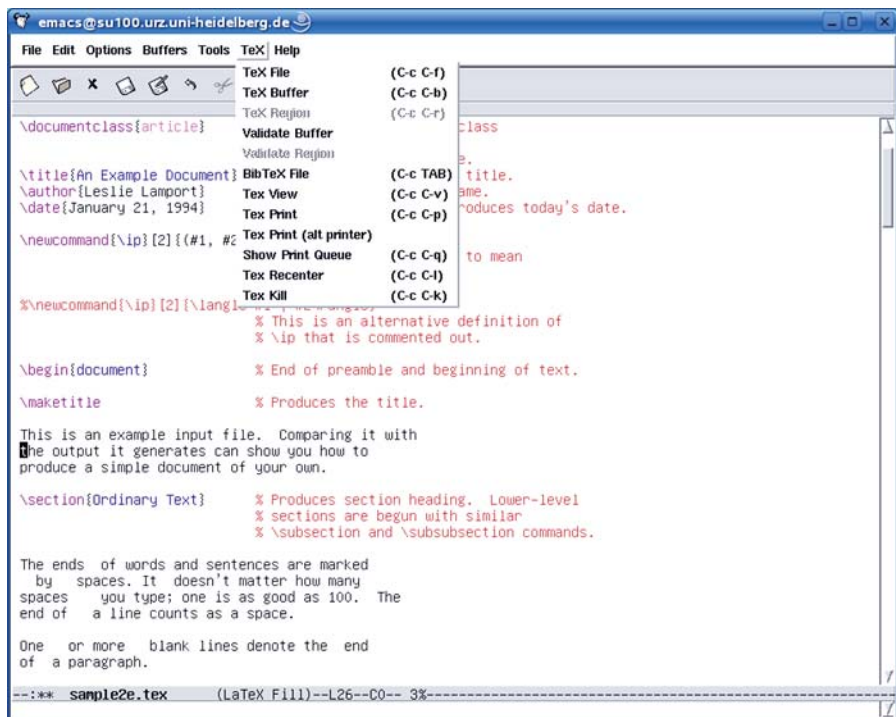
Im aktuellen Puffer kann man mit **Strg**–**s** ab der Cursorposition in Richtung des Puffer-Endes oder mit **Strg**–**r** in Richtung Puffer-Anfang suchen. Der gesuchte Text wird im Minipuffer eingegeben und mit **↵** abgeschlossen. Nach **Esc** **Strg**–**s** oder **Esc** **Strg**–**r** kann ein zu suchender Text auch als regulärer Ausdruck angegeben werden. In beiden Fällen kann mit **Strg**–**s** die Suche in der gleichen Richtung fortgesetzt oder mit **Strg**–**r** in der entgegengesetzten Richtung gesucht werden.

Zum Suchen und Ersetzen dienen **Esc** **%** oder bei regulären Ausdrücken **Esc** **Strg**–**%** . Der zu suchende Text und der Ersetzungstext werden wieder im Minipuffer eingegeben und jeweils mit **↵** abgeschlossen. Die nächste gefundene Stelle wird im Puffer angezeigt, und die Änderung kann mit der Taste **Leertaste** bestätigt oder mit **Backspace** unterdrückt werden. **?** informiert über diese beiden und weitere Steuerungseingaben an dieser Stelle. Nach der Steuerungseingabe wird die zugehörige Aktion ausgeführt und zur nächsten Fundstelle gewechselt.

Beim Bearbeiten umfangreicher Dokumente, die sich gegebenenfalls aus mehreren Dateien zusammensetzen, kann der Gebrauch von Lesezeichen (‚Bookmarks‘) hilfreich sein. Zum Anlegen, Verwalten und Anspringen von Lesezeichen dient in Emacs das Menü **Edit** → **Bookmarks**. Die Lesezeichen können auf Stellen in mehreren Puffern oder Dateien verweisen. Die gesetzten Lesezeichen können in einer Datei gespeichert und zu einem anderen Zeitpunkt zurück geladen werden.

14.1.1.6 LaTeX-Modus

Emacs erkennt an der Erweiterung **tex**, **ltx**, **cls** oder **sty** des Dateinamens, dass eine **T_EX**-Datei geöffnet wird, und aktiviert seinen **LaTeX**-Modus (oder den **T_EX**-Modus, wenn dieser im Konfigurationsmenü eingestellt wurde). In der Statuszeile wird dann in der runden Klammer **LaTeX** (oder **T_EX**) angezeigt.

Abbildung 14.2: Emacs beim Editieren einer $\text{\LaTeX}$ -Datei

In diesem Modus steht ein zusätzliches Pulldown-Menü **TeX** zur Verfügung. Über dieses Menü können wichtige Aktionen wie Formatieren, Ansehen oder Ausdrucken des Dokuments ebenso durchgeführt werden wie ein Aufruf von **BIB $\text{\TeX}$** . Abbildung 14.2 zeigt Emacs in diesem Modus beim Bearbeiten einer $\text{\LaTeX}$ -Datei mit aufgeklapptem Pulldown-Menü **TeX**. Die wichtigsten Aktionen werden im Folgenden vorgestellt.

TeX $\rightarrow$ **TeX File** (**Strg**–**c** **Strg**–**f**) speichert die im aktuellen Puffer bearbeitete $\text{\LaTeX}$ -Datei und formatiert sie.

TeX $\rightarrow$ **TeX Buffer** (**Strg**–**c** **Strg**–**b**) speichert den Inhalt des Puffers in einer temporären Datei und formatiert diese, ohne die dem Puffer zugeordnete Datei zu aktualisieren.

Ein markierter Bereich des Puffers lässt sich formatieren mit **TeX** $\rightarrow$ **TeX Region** (**Strg**–**c** **Strg**–**r**). Dazu wird eine temporäre Datei erzeugt, in die zunächst ein Standard-Prolog einschließlich des Befehls `\begin{document}` geschrieben wird. Anschließend wird der markierte Bereich gefolgt von einem Standard-Epilog bestehend aus `\end{document}` in die Datei geschrieben. Falls gewünscht können der Standard-Prolog

und -Epilog durch eigene Versionen ersetzt werden. Damit ist es beispielsweise möglich, benötigte Pakete zu laden oder eigene Definitionen einzubinden.

Die beim Formatieren erzeugte Log-Datei lässt sich in allen diesen Fällen mit `TeX → TeX Recenter` (`[Strg]-[c]` `[Strg]-[l]`) in einem eigenen Puffer anzeigen. Während des `TEX`-Laufes wird diese wie beim Aufruf von `LATEX` in einem Befehlsfenster angezeigt. Falls erforderlich kann ein gerade ausgeführter `LATEX`-Aufruf mit `TeX → TeX Kill` (`[Strg]-[c]` `[Strg]-[k]`) abgebrochen werden.

Auch der Aufruf von `BibTEX` über `TeX → BibTeX File` (`[Strg]-[c]` `[→]`) ist vorgesehen, ebenso wie `TeX → TeX View` (`[Strg]-[c]` `[Strg]-[v]`) zum Aufruf des ‚Previewers‘ und `TeX → TeX Print` (`[Strg]-[c]` `[Strg]-[p]`) zur Ausgabe auf dem Standarddrucker.

14.1.2 Emacs mit AUCT_EX

Emacs versteht sich als eine allgemeine Programmieroberfläche, in der `LATEX` nur eine unter vielen unterstützten ‚Programmiersprachen‘ ist. Der `LaTeX`-Modus stellt neben den allgemeinen Funktionen eines Editors auch die, bei der Arbeit mit `LATEX`-Dateien benötigten, Programmaufrufe zur Verfügung. In ähnlicher Weise werden auch die anderen ‚Programmiersprachen‘ unterstützt.

Moderne grafische Oberflächen bieten weitergehende Unterstützung beim Erstellen von `TEX`- und `LATEX`-Dokumenten. Häufig können wichtige Befehle und Umgebungen beispielsweise aus Pulldown-Menüs übernommen werden. Einige Oberflächen erläutern dabei auch ihre Verwendung und die Bedeutung der Argumente. Das Emacs-Paket AUCT_EX stellt in Emacs Programmierumgebungen für `TEX` und `LATEX` zur Verfügung, die mit den grafischen Oberflächen vergleichbare Fähigkeiten aufweisen.

Ausgehend vom `TeX`-Modus des Emacs entwickelte Lars Peter Fischer 1986 erste Funktionen zum Einfügen von Makros zur Fontauswahl und zur Eingabe dänischer Sonderzeichen. So entstand die erste – noch sehr einfache – Version von AUCT_EX. An der weiteren Entwicklung beteiligte sich neben Per Abrahamsen eine große Anzahl weiterer Entwickler. Die erste öffentliche Version stellte 1991 Kresten Krab Thorup zur Verfügung, der bis einschließlich Version 6.0 im Jahr 1992 Hauptautor und Koordinator war.

Die ursprünglichen Fähigkeiten wurden dabei so erweitert, dass heute mit AUCT_EX eine `TEX`-Oberfläche zur Verfügung steht, die sich mit guten

grafischen Oberflächen durchaus messen kann. Befehle und Umgebungen können auch hier über Pulldown-Menüs eingefügt werden.

Während bei grafischen Oberflächen die Maus eine wesentliche Rolle beim Einfügen von Befehlen und Umgebungen oder der Durchführung von Aktionen wie Formatieren und Ansehen des Dokuments spielt und lediglich teilweise durch Tastaturbefehle ergänzt wird, steht bei Emacs mit AUCTeX die Tastatureingabe im Vordergrund. Dadurch entfällt die Notwendigkeit, beim Schreiben eines Dokuments ständig zwischen Tastatur und Maus zu wechseln. Die Pulldown-Menüs von Emacs sind aber in jedem Fall eine große Hilfe, da die Tastaturkürzel in ihnen angezeigt werden.

14.1.2.1 Konfiguration

Nach Installation des Emacs-Pakets AUCTeX kann es nach Aufruf von Emacs geladen werden. In der Regel wird man das Paket aber nicht jeweils erneut manuell laden wollen, also fügt man in die Konfigurationsdatei `~/.emacs` die Zeile

```
(require 'tex-site)
```

ein. Die weitere Konfiguration kann anschließend beim Editieren einer – gegebenenfalls leeren – TeX-Datei über die Konfigurationsmenüs von Emacs in der Gruppe AUCTeX erfolgen. Das Öffnen einer TeX-Datei ist übrigens erforderlich, da AUCTeX nur bei Bedarf geladen wird.

Über **Options** → **Customize Emacs** → **Specific Group...** und Eingabe des Gruppennamens AUCTeX im Minipuffer (unterste Zeile eines Emacs-Fensters) oder noch schneller über **LaTeX** → **Customize AUCTeX** → **Browse Options** gelangt man direkt zum Konfigurationsmenü von AUCTeX. In einer ganzen Reihe von Untermenüs lassen sich die verschiedenen Teilbereiche von AUCTeX den Wünschen des Anwenders anpassen. Beispielsweise können in der Gruppe **Tex Command** die über das Pulldown-Menü **Command** durchgeführten Programmaufrufe verändert oder neue Programmaufrufe hinzugefügt werden.

Eine manuelle Anpassung der verfügbaren Befehle und Umgebungen ist übrigens nicht sinnvoll, da AUCTeX ohnehin alle ihm bekannten Verzeichnisse mit TeX-Quellen analysiert und die Informationen zu jeder Quelldatei in Form einer Lisp-Datei ablegt. Sobald ein Paket oder eine Quelldatei eingebunden wird, verwendet AUCTeX die zugehörige Informationsdatei zur Aktualisierung der verfügbaren Befehle und Umgebungen. Erforderlich ist daher allenfalls die Anpassung der Liste der Suchpfade für TeX-Quelldateien in den Optionen **Tex Macro Global** und **Tex Macro Private** in der Gruppe **Tex File**.

14.1.2.2 Gemeinsamkeiten und Unterschiede

AUCTeX ist kein eigenständiges Programm, sondern eine Erweiterung von Emacs. TeX- und LaTeX-Modus von ProgEmacs werden jeweils durch einen eigenen erweiterten Modus ersetzt. Die in 14.1.1.5 beschriebenen Editierfunktionen können ohne Einschränkung auch unter AUCTeX genutzt werden. Die Pulldown-Menüs **File**, **Edit**, **Options**, **Buffers**, **Tools** und **Help** stellen die gewohnten Einträge zur Verfügung. Das Editieren einer L^AT_EX-Datei erfolgt also in der von Emacs gewohnten Weise.

Ohne AUCTeX unterstützt Emacs die Arbeit mit L^AT_EX-Dateien durch das Pulldown-Menü **TeX**. Über dieses Menü können L^AT_EX-Dokumente formatiert, betrachtet und gedruckt werden. AUCTeX ersetzt das Menü in seinem LaTeX-Modus durch das im Umfang erweiterte Pulldown-Menü **Command**, das unten in Abschnitt 14.3 beschrieben ist.

Die wesentliche Erweiterung ist jedoch das Pulldown-Menü **LaTeX**. Über dieses Menü können Umgebungen durch Auswahl in die L^AT_EX-Datei übernommen werden. Ein wesentlicher Vorteil von AUCTeX ist dabei, dass es die Klassen und Pakete in den Standardverzeichnissen und sogar eigene Eingabedateien analysiert und die in ihnen definierten Befehle und Umgebungen erkennt. Die gefundenen Umgebungen werden in die Untermenüs von **LaTeX** → **Insert Environment** und **LaTeX** → **Change Environment** aufgenommen. Diese Untermenüs dienen lediglich dazu, das Pulldown-Menü nicht zu lang werden zu lassen. Es handelt sich dabei um einzelne Abschnitte einer alphabetisch sortierten Liste.

Eine entsprechende Liste von Befehlen wird nicht aufgebaut. Selbst die Aufteilung in Abschnitte könnte nicht verhindern, dass entweder die Liste der Abschnitte oder die Abschnittslisten zu umfangreich würden. Statt dessen bietet AUCTeX die Möglichkeit, nach Eingabe des Anfangs eines Befehls diesen zu vervollständigen.

Selbstdefinierte Befehle und Umgebungen werden allerdings nur dann unterstützt, wenn AUCTeX dementsprechend konfiguriert ist. Falls das noch nicht der Fall ist, kann die Konfiguration über das Konfigurations-Menü erfolgen, das direkt über **LaTeX** → **Customize AUCTeX** → **Browse Options** erreichbar ist. Die gesuchten Optionen **Tex Parse Self** und **Tex Auto Save** sind in der Gruppe **Tex Parse** zu finden und müssen beide den Wert **on** haben. Zum Umschalten dient der Button **Toggle**. Nach Änderungen ist entweder bei der geänderten Einstellung (nur für genau diese Änderung) oder am Anfang der Gruppe für alle Änderungen in der Gruppe über den Button **State** die Änderung zu aktivieren. Dies kann zum Testen durch Auswahl von **Set for Current Session** nur für die aktuelle Emacs-Sitzung oder alternativ **Save for Future Sessions** als dauerhafte Änderung gespeichert werden. Die gleiche Wirkung wie diese beiden

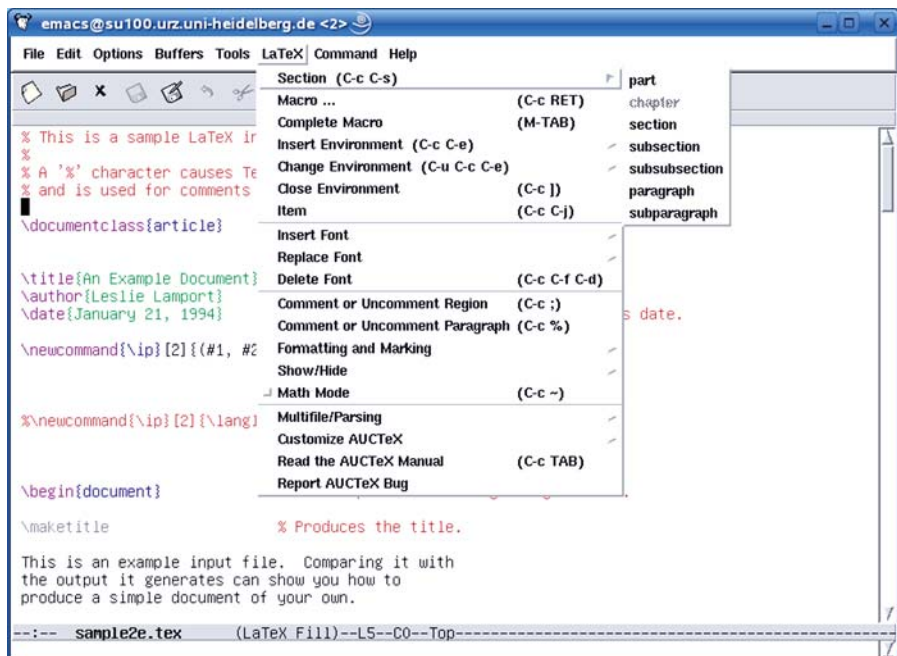


Abbildung 14.3: AUCTeX-Menü LaTeX mit Untermenü Section

Menü-Einträge haben – für die ganze Gruppe – auch die gleichnamigen Buttons am Anfang des Puffers.

14.1.2.3 Erstellen und Bearbeiten von L^AT_EX-Dateien

Die in 14.1.1.5 beschriebenen Editierfunktionen bilden auch bei Verwendung von AUCTeX die Grundlage des Editierens. Zusätzlich verfügt AUCTeX über das Pulldown-Menü LaTeX, über das mit der Maus nahezu alle L^AT_EX-Befehle und -Umgebungen in den Text eingefügt werden können. Abbildung 14.3 zeigt AUCTeX beim Bearbeiten eines Artikels (Dokumentklasse article) mit geöffnetem Menü LaTeX und darin geöffnetem Untermenü Section. Angeboten werden die in dieser Dokumentklasse verfügbaren Überschriften-Befehle.

14.1.2.4 Formatieren von L^AT_EX-Dateien und andere Programmaufrufe

In Command sind die beim Erstellen eines Dokuments mit Hilfe von L^AT_EX und seinen Zusatzprogrammen benötigten Programmaufrufe zu finden.

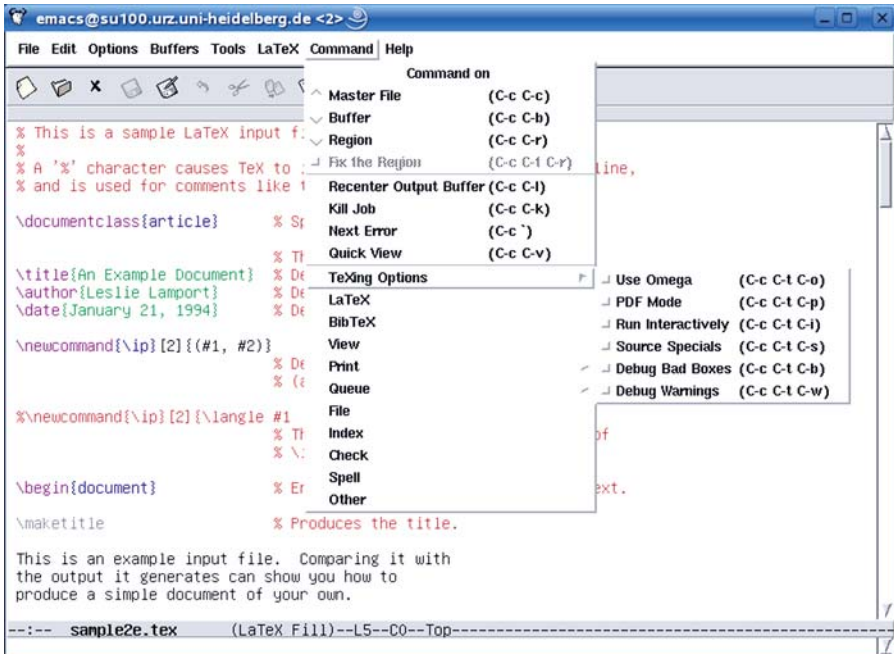


Abbildung 14.4: AUCTeX-Menü Command mit Untermenü TeXing Options

Daneben finden sich hier auch ‚Radio Buttons‘, um die Formatierung zu steuern. Man kann wählen, ob die Hauptdatei („Master File“), der (gesamte) Puffer oder nur ein markierter Bereich im Puffer formatiert werden soll. Unter **TeXing Options** kann neben anderem auch festgelegt werden, dass statt des klassischen $\text{\TeX}$, $\text{\pdfTeX}$ oder Ω (Omega) zum Formatieren verwendet wird. Abbildung 14.4 zeigt das Menü **Command** mit geöffnetem Untermenü **TeXing Options**.

Der zweite Block in diesem Menü unterstützt das Formatieren. Notfalls kann über den Eintrag **Kill Job** ($\text{\Strg} - \text{\c} \text{\Strg} - \text{\k}$) ein aufgerufenes Programm abgebrochen werden. Die Ausgaben aufgerufener Programme – etwa von $\text{\TeX}$ zum Formatieren – werden in einem Ausgabepuffer („Output Buffer“) angezeigt. Über den Eintrag **Recenter Output Buffer** ($\text{\Strg} - \text{\c} \text{\Strg} - \text{\l}$) lässt sich dieser Puffer parallel zur bearbeiteten $\text{\LaTeX}$ -Datei anzeigen. Normalerweise wird stets das Ende dieses Puffers mit den aktuellen Ausgaben angezeigt. Wurde allerdings zurückgeblättert, etwa um bestimmte Ausgaben genauer in Augenschein zu nehmen, kann man über diesen Eintrag zu den aktuellen Ausgaben am Ende des Puffers wechseln. Der Eintrag **Quick View** ($\text{\Strg} - \text{\c} \text{\Strg} - \text{\v}$) öffnet einen Previewer (in der Regel $\text{\xdvi}$) mit der zuletzt formatierten $\text{\LaTeX}$ -Datei.

Der letzte Block des Menüs enthält Programmaufrufe, die bei der Arbeit mit $\text{\LaTeX}$ wichtig sind:

- Der Eintrag **LaTeX** formatiert – je nach Einstellung der ‚Radio Buttons‘ – die Hauptdatei, den Puffer oder den ausgewählten Bereich des Puffers.
- **BibTeX** ruft $\text{\BIBTeX}$ auf.
- **View** dient zum Aufruf des Previewers, wobei im Unterschied zu **Quick View** der Befehl zum Aufruf im Minipuffer angezeigt wird und verändert werden kann.
- **Print** öffnet ein Untermenü mit Einträgen für verschiedene Drucker. Standardmäßig ist neben einem voreingestellten Drucker ein Eintrag für andere Drucker vorgesehen, bei dem der gewählte Drucker im Minipuffer eingegeben werden kann. Das analoge Untermenü **Queue** erlaubt das Abfragen der Druckaufträge.
- **File** ruft **dvips** auf, um eine PostScript-Ausgabe zu erstellen; der Befehl zum Aufruf wird im Minipuffer angegeben und lässt sich verändern.
- Über **Index** kann **MakeIndex** aufgerufen werden.
- **Check** ruft das Programm **lacheck** auf, um zu prüfen, ob die Eingabedateien korrekte $\text{\LaTeX}$ -Dateien sind.
- Die Rechtschreibung lässt sich mit **Spell** überprüfen. Verwendet wird dazu in der Regel das Programm **ispell**.
- Über **Other** können andere Programme durch Eingabe des auszuführenden Befehls im Minipuffer aufgerufen werden.

Über die vorgestellte Konfiguration von $\text{\AUCTeX}$ können die voreingestellten Befehle verändert oder zusätzliche Programmaufrufe hinzugefügt werden.

14.2 Grafische Entwicklungsumgebungen

Eine integrierte Entwicklungsumgebungen (IDE) unterstützt den Autor beim Erstellen und Setzen seines Texts mit $\text{\LaTeX}$ oder $\text{\TeX}$. IDEs stehen für fast alle Betriebssysteme zur Verfügung. Wir beschränken uns auf die folgenden IDEs:

- $\text{\TeX}$ maker für die Betriebssysteme Linux und Windows,

- $\text{\TeX}$ nicCenter für das Betriebssystem Windows und
- Kile für das Betriebssystem Linux.

Grafische Entwicklungsumgebungen für den MacIntosh wären sicherlich auch interessant, wir möchten uns jedoch auf die obigen beschränken, da sie die zur Zeit komfortabelsten sind und außerdem keine Gebühren oder sonstige Kosten bei ihrer Verwendung anfallen.

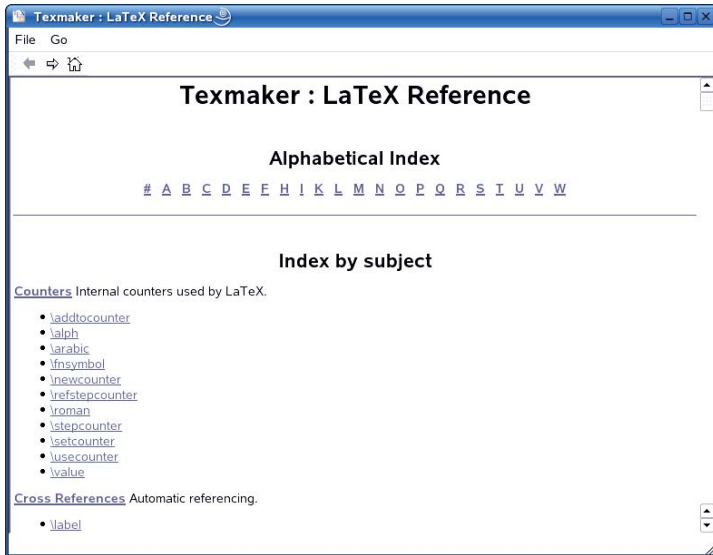


Abbildung 14.5: $\text{\TeX}$ maker $\text{\LaTeX}$ -Referenz

Generell ist bei allen drei Entwicklungsumgebungen eine ausgezeichnete $\text{\LaTeX}$ -Referenz integriert, mit der die Syntax jedes Befehls online nachgeschlagen werden kann.

Alle Werkzeuge, die zum Erstellen erforderlich sind, stehen innerhalb der IDE zur Verfügung. Vorhanden sind insbesondere `tex` und `latex` zur Formatierung einer $\text{\TeX}$ - bzw. $\text{\LaTeX}$ -Eingabe sowie `pdflatex` zur Verarbeitung eines $\text{\LaTeX}$ -Texts mit PDF-Ausgabe. Die Ausgabe, standardmäßig ist das eine DVI-Datei, kann in die unterschiedlichsten Formate konvertiert werden. Auch zur Erstellung eines Indexes bzw. einer Bibliografie sind die entsprechenden Routinen einfach mit einem Mausklick erreichbar. Die genannten IDEs gehen davon aus, dass unter Linux ein `te $\text{\TeX}$` und unter Windows das Paket `Mik $\text{\TeX}$` installiert sind. Diese beiden Software-Pakete kann man bequem und natürlich kostenlos vom CTAN-Server von DANTE (deutschsprachige Anwendervereinigung $\text{\TeX}$

e. V.) <http://www.dante.de> herunterladen oder auch direkt vom Server installieren.

Für das Erstellen eines $\text{\LaTeX}$ -Dokuments mit einer IDE sind die folgenden Punkte bedeutsam:

- Ein Editor für die Eingabe ist integriert.
- Die Verarbeitung mit $\text{\LaTeX}$ kann gestartet werden.
- Die Ausgabe (Log-Datei) wird in ein Fenster angezeigt und analysiert; man kann bequem mit einem Mausklick von Fehler zu Fehler springen.
- Oft ist ein *Quickstart* vorhanden, der je nach Programmierung Verarbeitung (auch mit Zwischenschritten wie Bibliografie- und Index-Generierung) und Anzeige bzw. Ausgabe in einem Schritt durchführt.

Sind diese Punkte erfüllt ist es recht bequem ein Dokument zu erstellen, zu bearbeiten und die Fehler, die bei befehlsgesteuerter Texterstellung unvermeidlich sind, zu analysieren und zu beseitigen.

Das Betrachten der formatierten $\text{\LaTeX}$ -Ausgabe muss ebenso leicht möglich sein. Entsprechend dem generierten Ausgabeformat (normalerweise eine DVI- oder PDF-Datei, eventuell ist die DVI-Datei schon nach PostScript konvertiert), wird das entsprechende integrierte oder externe Programm aufgerufen:

- eine DVI-Datei (device independent) kann mittels eines DVI-Previewer angesehen werden,
- bei einer PS-Datei (PostScript) wird ein PostScript-Viewer, z. B. Ghostview oder KGhostview gestartet, und
- für eine PDF-Datei (portable document format) wird meist der Acrobat Reader von Adobe Inc. eingesetzt. Unter einem modernen Linux wird anstelle des Acrobat Readers oft das Programm KPDF verwendet (automatisch bei Verwendung der KDE-Oberfläche), welches kleiner, schneller und handlicher ist als die neue Version 7 des Acrobat Readers.

Der in der IDE integrierte Editor bietet jegliche Unterstützung, die man heute erwarten kann:

- Befehlskennzeichnung,
- Rechtschreibprüfung in unterschiedlichen Sprachen,
- dynamischer Zeilenumbruch,
- unbegrenztes Zurücknehmen und Wiederholen des letzten Befehls, sowie der Sprung zur letzten Änderung,

weiteren Konstrukten zu helfen. Damit braucht man lediglich noch anzuklicken, was man möchte, eventuell ein paar Werte einzutragen und der $\text{\LaTeX}$ -Code wird dann automatisch generiert. Hier einige Beispiele:

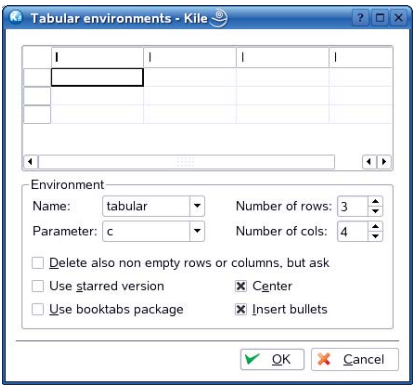


Abbildung 14.7: Tabellen-Editor

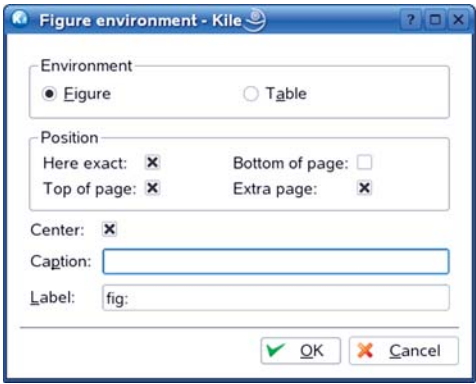


Abbildung 14.8: Erzeugen von Gleitobjekten

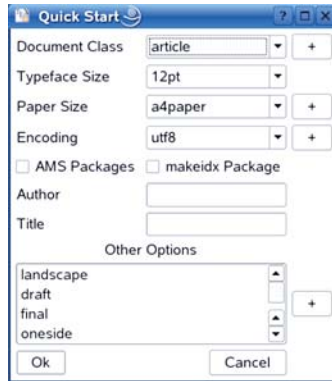
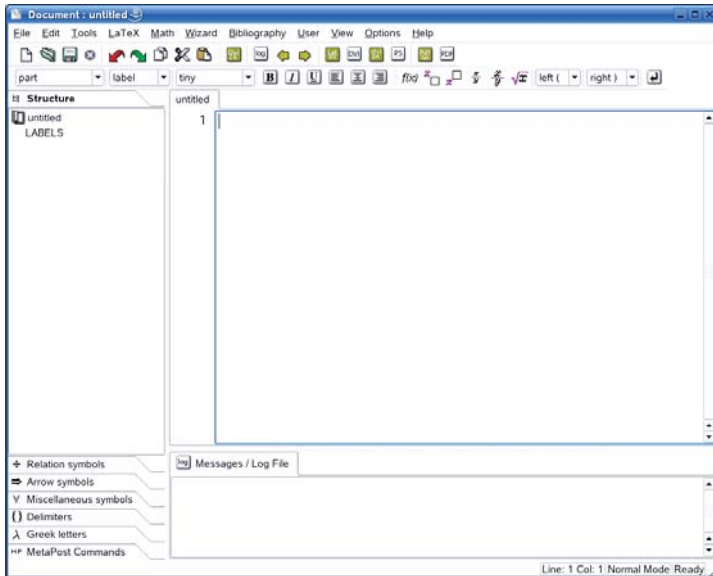


Abbildung 14.9: Quickstart beim Öffnen einer Datei

14.2.1 T_EXmaker

T_EXmaker ist die Benutzeroberfläche, die zusammen mit T_EXnicCenter am Längsten verfügbar ist. T_EXmaker ist außerdem die einzige Umgebung, die es für verschiedene Betriebssysteme gleichzeitig gibt.

Abbildung 14.10: T_EXmaker-Oberfläche

Zuerst nur für das Betriebssystem Linux entwickelt, sind neben dem Quellcode, den man selbst übersetzen kann, lauffähige Pakete für die unterschiedlichen Linux-Versionen (texmaker-1.12.spec) verfügbar. Ferner sind Versionen für Mac OS X und Windows vorhanden. Man kann die benötigten Dateien herunterladen unter <http://www.xmlmath.net/texmaker>. Die Installation gelingt ohne größere Probleme. Falls man den Quellcode selbst übersetzen möchte, sollte man nicht vergessen darauf zu achten, dass das *Qt toolkit* in der Version 3.2 oder neuer installiert ist.

14.2.2 T_EXnicCenter

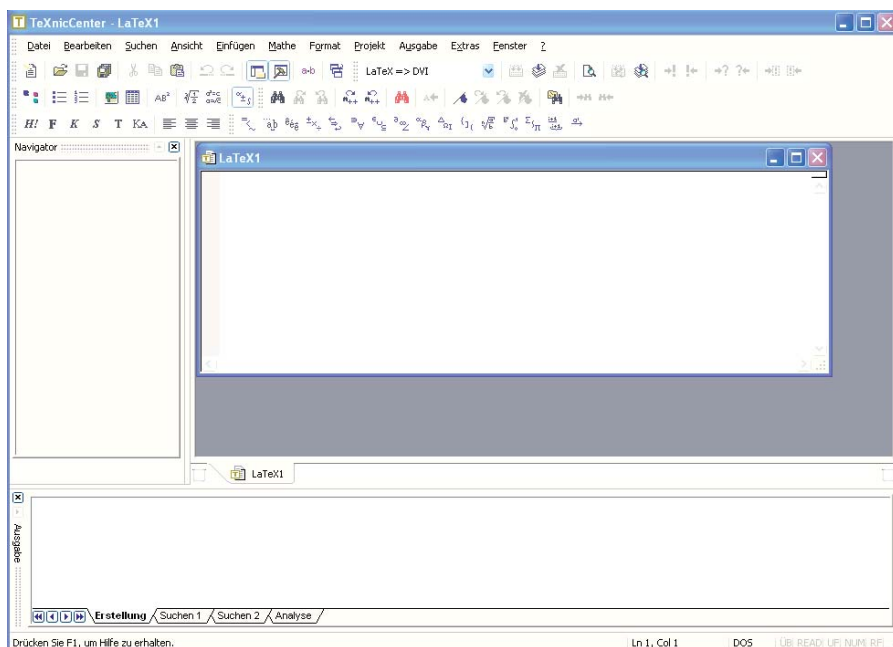


Abbildung 14.11: T_EXnicCenter-Oberfläche

T_EXnicCenter ist eine spezielle integrierte Entwicklungsumgebung, um L^AT_EX-Dokumente unter Microsoft Windows (Windows 9x/ME, NT, 2000 und XP) und ausschließlich dort zu erzeugen (eine Portierung in anderen Betriebssysteme ist nach Aussage der Entwickler nicht möglich). Hier ist das Ausgabemenü frei konfigurierbar (auf Grund seiner Konstruktion wären theoretisch unendlich viele Einträge möglich – nur der Bildschirm

würde beim Aufklappen zu klein), so sind eigenen Vorstellungen keine Grenzen gesetzt.

Der Editor ist voll konfigurierbar, und – als zusätzliches kleines Bonbon – seine Menü-Sprache (wie auch die des kompletten $\text{\TeX}$ nicCenter) ist Deutsch.

Interessant ist auch die Möglichkeit, eigene Vorlagen für entsprechende Dokumente vorzuhalten. Für die Generierung der Ausgabe kann man gemäß den eigenen Bedürfnissen Profile anlegen.

14.2.3 Kile

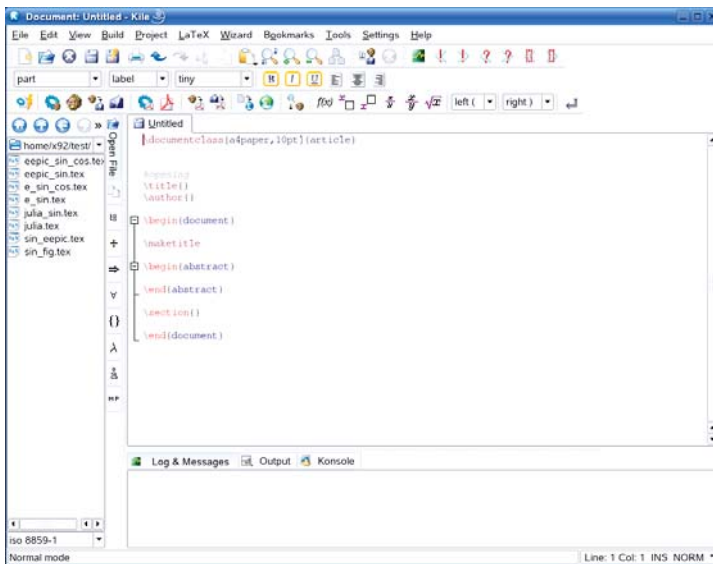


Abbildung 14.12: Kile-Oberfläche

Kile ist der Nachfolger von $\text{\TeX}$ maker2, welcher wiederum auf $\text{\TeX}$ maker fußt. Der Funktionsumfang hat sich jedoch immens gesteigert. Man kann Kile als den *Rolls Royce* unter den Entwicklungsumgebungen betrachten. Allerdings muss man die neuesten Versionen (1.7 oder neuer) – aktuell ist die Version 1.8.1 – einsetzen, die wiederum ein aktuelles KDE (3.3 oder neuer) erfordern. Dies macht eine Portierung in andere Betriebssysteme sehr schwierig – wenn auch nicht unmöglich. Generell ist zu sagen, man benötigt am Besten eine neue Linux Installation.

Alle Glanzlichter aufzuzählen ist fast unmöglich, einiges darf jedoch nicht unerwähnt bleiben. Bereits am Anfang macht man Bekanntschaft mit dem Dokument-Assistenten, der einen schnellen Anfang ermöglicht. $\LaTeX$ - und $\TeX$ -Befehle werden automatisch vervollständigt, wobei immer alle möglichen Varianten im Auswahlménü angeboten und durch Mausklick ausgewählt werden können. Umgebungen werden automatisch abgeschlossen.

Der integrierte Editor kann durch *Plugins* in seiner Funktionalität erweitert werden. Dadurch kann eine Rechtschreibprüfung (soweit sie installiert ist), die automatische Vervollständigung von Wörtern in Dokumenten und Weiteres hinzugefügt werden.

14.2.4 Fazit

Die Zeit, in der man $\LaTeX$ -Dokumente mit einem simplen Editor erstellt hat, ist vorüber. Auch wenn die integrierten Entwicklungsumgebungen (IDE) für $\TeX$ / $\LaTeX$ kein echtes WYSIWYG (*what you see is what you get*) realisieren – inwieweit dies Office bzw. Open-Office-Produkte können, sei dahingestellt – kann man zu jeder Zeit die Formatierung (durch einen einfachen Mausklick) nachprüfen. Unter Linux kann man auch ein Ausgabefenster permanent geöffnet lassen, um so bequem nach dem Aufruf der Textformatierung (auch ein Mausklick) das Ergebnis sehen zu können.

Unter den drei behandelten Entwicklungsumgebungen einen Favoriten zu benennen ist nicht ganz einfach. Als Linux-Anwender, der unter einem neuen KDE arbeitet, ist Kile die erste Wahl. Unter Windows wird man sich wahrscheinlich für $\TeX$ nicCenter entscheiden. $\TeX$ maker wäre überlegenswert, wenn man zwischen verschiedenen Betriebssystemen wechselt und nur eine IDE verwenden möchte.

Generell hat jedoch die (eigentlich noch nie berechnigte) Angst vor der Komplexität von $\LaTeX$ ihre Grundlage verloren.

14.3 WYSIWYG-ähnliche Oberflächen

Ziel der Entwicklung von $\TeX$ war es, zunächst die Arbeit eines Buchsetzers per EDV zu ermöglichen und darüber hinaus weitestgehend zu automatisieren. Da einem Setzer ein fertiges Dokument vorliegt, kann er den Satz von Absätzen, Seiten und dem gesamten Text optimieren, so dass eine optimale Seitenfüllung erreicht wird. Gut lesbare Texte haben

eine optimale Seitenfüllung, denn störende weisse Flüsse (die das Auge anziehen) werden vermieden. $\text{T}_{\text{E}}\text{X}$ und damit auch $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ folgen dieser Strategie.

Die insbesondere unter Windows weit verbreiteten Textverarbeitungssysteme folgen im Gegensatz dazu der WYSIWYG-Strategie: Bereits bei der Eingabe wird der Text genau so angezeigt, wie er später gedruckt wird (WYSIWYG = ‚What You See Is What You Get‘).

Aufgrund dieser unterschiedlichen Strategien ist es kaum möglich, eine echte WYSIWYG-Oberfläche für $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ zu entwickeln. Ein weiteres Problem mit WYSIWYG-Oberflächen besteht darin, dass kleine Änderungen zum Umformatieren ganzer Absätze führen können; das könnte im Fall einer WYSIWYG-Oberfläche zu sehr unruhiger Darstellung eines eingegebenen Texts führen. Zumindest in solchen Fällen ist eine WYSIWYG-Oberfläche sicher kaum wünschenswert.

Nicht zuletzt aufgrund dieser Schwierigkeiten gibt es bisher keine echten WYSIWYG-Oberflächen. Die hier vorgestellten Programme $\text{L}_{\text{y}}\text{X}$ und $\text{T}_{\text{E}}\text{X}$ -macs werden daher auch nicht als WYSIWYG, sondern als WYSIWYG-ähnliche Oberflächen bezeichnet.

Beide Programme benutzen intern eigene Darstellungen der Dokumente, die unabhängig von $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ sind. Der Import von $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Dateien wird von beiden unterstützt. $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Befehle können in Dokumente eingefügt werden; die durch sie erzeugten Ausgaben können in der Regel nicht angezeigt werden. Die Ausgabe der Dokumente im $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Format wird unterstützt. Ebenfalls möglich ist der Export ins HTML- oder XML-Format. Für das Aufbereiten der Druckausgabe wird $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ verwendet.

Die Bedienung der Programme über Pulldown-Menüs und Icons ist ähnlich. Soweit nur die vorgesehenen Textelemente verwendet werden, können Dokumente auch ohne $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Kenntnisse in der von $\text{T}_{\text{E}}\text{X}$ gewohnten Qualität erstellt werden. Bei Bedarf ist es möglich, $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Befehle einzufügen.

Deutliche Unterschiede gibt es jedoch bei den Tastaturbefehlen: Bei $\text{L}_{\text{y}}\text{X}$ entsprechen sie weitgehend denen der üblichen Editoren unter KDE. Für $\text{T}_{\text{E}}\text{X}$ macs wurden die Tastaturbefehle des Emacs übernommen.

14.3.1 $\text{L}_{\text{y}}\text{X}$

Das hochinteressante Programm $\text{L}_{\text{y}}\text{X}$ von Matthias Ettrich gehört zu den WYSIWYG-ähnlichen $\text{T}_{\text{E}}\text{X}$ -Oberflächen. Der Anwender kann ohne $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Befehle seinen Text eingeben, für den Preview und zur Druckausgabe verwendet $\text{L}_{\text{y}}\text{X}$ $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ als hochwertigen Formatierer und die $\text{T}_{\text{E}}\text{X}$ -Programme zur Ausgabe.

Die grafische Oberfläche des Programms basiert wahlweise auf dem XForms-Toolkit oder auf Gnome und QT2. Da KDE unter Linux ebenfalls auf Gnome und QT aufsetzt, steht die aktuelle LyX-Version auch unter KDE ohne Verzögerung zur Verfügung; ein Rückgriff auf KLyX – eine Anpassung der veralteten LyX-Version 0.12.0 an KDE – erübrigt sich damit.

Das Arbeiten mit LyX ist ohne L^AT_EX-Kenntnisse möglich. In Anlehnung an die Abkürzung ‚WYSIWYG‘ wird LyX in der Dokumentation auch als WYSIWYM-Oberfläche bezeichnet (‚What You See Is What You Mean‘), da LyX beim Schreiben den Text ähnlich wie in der späteren Druckausgabe darstellt. Abweichungen von der Druckfassung beruhen zum einen auf der Anpassung an die Bildschirmausgabe – beispielsweise Anpassung der Zeilenlänge an die Breite des geöffneten Fensters, fehlende Seitenumbrüche, Unterdrücken der Anzeige von Verzeichnissen, Fußnoten und Randnotizen im Text – und zum anderen auf der Tatsache, dass die Bildschirmdarstellung programmintern aufgebaut wird und nicht auf einer Formatierung mit L^AT_EX beruht.

Auf der einen Seite bietet LyX die Vorteile anderer weit verbreiteter Textverarbeitungssysteme am PC, wie einfache Bedienung über Icons und Menüs und die direkte Darstellung am Bildschirm. Andererseits druckt LyX Dokumente in der von L^AT_EX gewohnten hohen Qualität und nutzt die Vorteile von L^AT_EX: automatisch erstellte Inhalts-, Abbildungs- und Tabellenverzeichnisse, automatische Nummerierung von Überschriften, Tabellen, Abbildungen, Formeln und Fußnoten, automatisch aktualisierte Quer- und Literaturverweise und die Exaktheit bei der Darstellung mathematischer Formeln. Am Bildschirm werden die Verzeichnisse nur als Symbole angezeigt, ausgenommen das Inhaltsverzeichnis, das als eigenes Fenster geöffnet werden kann, mit dem Nutzen, darüber zu einem bestimmten Abschnitt zu wechseln. Alle Verweise haben in LyX die Bedeutung von Hyperlinks, über die man durch Anklicken mit der Maus zu der betreffenden Stelle wechseln kann.

Betrachtet man LyX als eine T_EX-Oberfläche, so ermöglicht sie das Schreiben von L^AT_EX-Dokumenten ohne L^AT_EX-Kenntnisse. Manuell erstellte Literaturverweise werden ebenso unterstützt wie mit BIB_TE_X erstellte Literaturdatenbanken. Literaturverweise und Verweise auf markierte Textelemente werden durch Angabe einer zugeordneten Nummer (beispielsweise einer Formelnummer oder der Nummer des Abschnitts) oder der Seite, auf der sie zu finden sind, automatisch auf dem Laufenden gehalten. Beim Einfügen von Verweisen bietet LyX in einem Pulldown-Menü alle im Dokument vereinbarten Marken an, aus denen diejenige ausgewählt werden kann, auf die verwiesen werden soll. Nicht zuletzt ist die automatische Positionierung von Textelementen entsprechend einem ausgewählten Layout ein wesentlicher Vorteil von L^AT_EX und damit auch

von LyX . Bei der Arbeit mit LyX kann man sich also ganz auf die inhaltlichen Aspekte konzentrieren und die formalen Aspekte LyX bzw. $\text{\LaTeX}$ überlassen.

Die größtenteils deutschsprachige Dokumentation umfasst rund 400 Seiten und führt den Benutzer in alle wesentlichen Funktionalitäten des Programms ein. Sie besteht aus einer Reihe von Dokumenten, die gleichzeitig als Online-Hilfe zur Verfügung stehen und Teil der Distribution sind. Die in der Dokumentation verwendeten Konventionen werden in einer *Einführung in LyX* , LyX Team (2003b) beschrieben. Eine praktische Einführung mit Übungsvorschlägen ist *Das LyX -Tutorium*, LyX Team (2003a). Als zentrale Referenz dient *Das LyX -Benutzerhandbuch*, LyX Team (2004a). Ergänzende Hinweise kann man in *Profi-Tips: erweiterte Textbearbeitung mit LyX* , LyX Team (2004b) finden. Die internen Funktionen, auf die man auch als Benutzer zugreifen kann, sind im *LyX Reference Manual*, LyX Team (1997) beschrieben.

Unter Linux wird LyX in der Regel als fertiges Paket angeboten, das in der üblichen Weise installiert werden kann. Dabei wird auch ein Eintrag in das Programm-Menü angelegt, um den Aufruf zu vereinfachen. Auf anderen Systemen wird man in der Regel die auf der weitgehend frei verfügbaren Forms-Bibliothek XForms basierende Installation wählen. In diesem Fall muss die XForms-Bibliothek bereits installiert sein, bevor LyX installiert werden kann. Ein Eintrag in ein geeignetes Oberflächen-Menü kann anschließend manuell vorgenommen werden. In einem Textfenster erfolgt der Aufruf am Besten mit folgendem Befehl:

```
lyx
```

Obwohl für heutige $\text{T}_{\text{E}}\text{X}$ -Version echte WYSIWYG-Oberflächen nicht möglich sind, kommt LyX diesem Ziel – für $\text{\LaTeX}$ – ziemlich nahe. Auf der einen Seite kann man mit LyX Standard-Dokumente ohne $\text{\LaTeX}$ -Kenntnisse erstellen. Andererseits erlaubt LyX das Einfügen von $\text{\LaTeX}$ -Befehlen und -Umgebungen, wobei diese Teile nur als Befehle und nicht WYSIWYG-mäßig dargestellt werden. Gegebenenfalls kann auch die $\text{\LaTeX}$ -Präambel des Dokuments editiert werden, etwa um $\text{\LaTeX}$ -Pakete mit `\usepackage`-Befehlen zu laden.

Abstriche am WYSIWYG-Konzept müssen auch beim Zeilen- und Seitenumbruch in Kauf genommen werden, da die hier geforderten Fähigkeiten von $\text{T}_{\text{E}}\text{X}$ es erfordern würden, weite Teile von $\text{T}_{\text{E}}\text{X}$ nachzuprogrammieren. Im Vertrauen auf die hohe Qualität von $\text{\LaTeX}$ wurde auf diesen Aufwand verzichtet.

Einige Unterschiede zu einem WYSIWYG-System zeigt Abbildung 14.13. Hier ist der Anfang der Einführung zu LyX in der WYSIWYM-Oberfläche oben und darunter ein Ausschnitt mit dem Beginn des ersten Kapitels

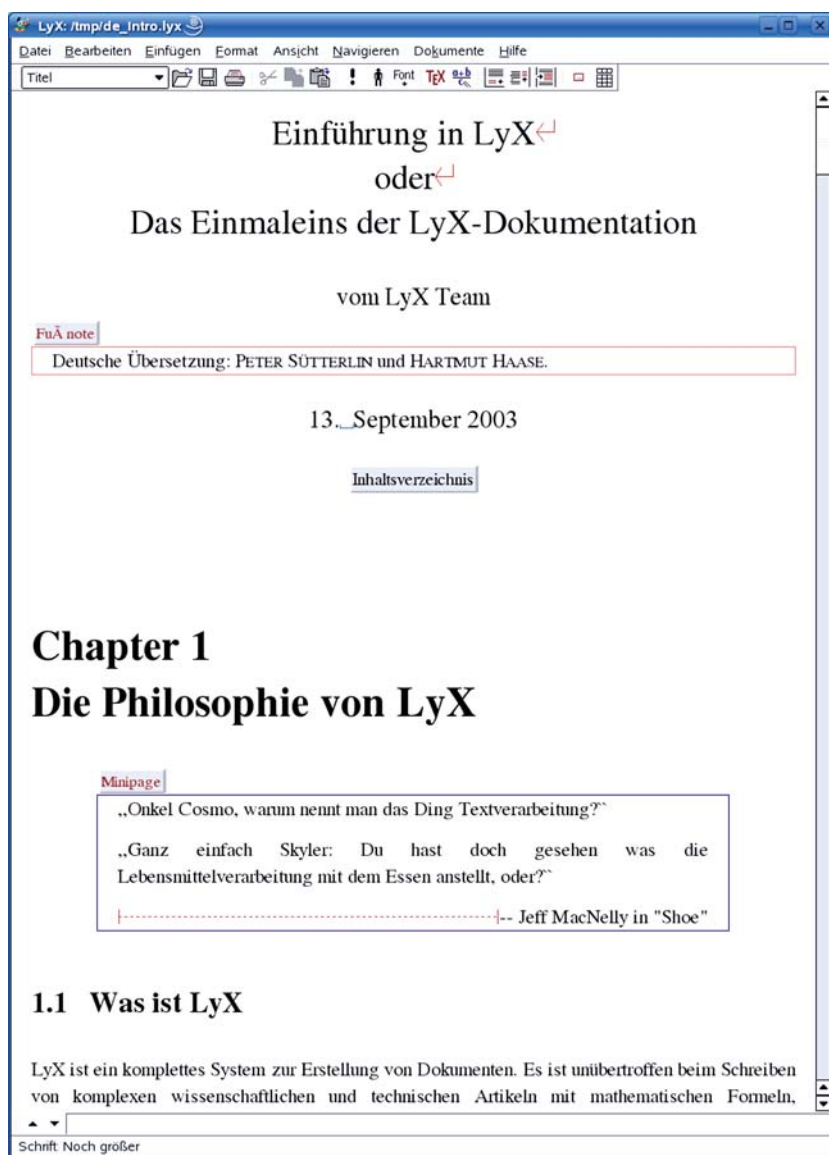


Abbildung 14.13: LyX-Oberfläche

aus dem gedruckten Dokument zu sehen. Eine vollständige Wiedergabe des in der Oberfläche gezeigten Ausschnitts ist nicht abgedruckt, da die Titelseite mit der zugehörigen Fußnote und das Inhaltsverzeichnis auf eigenen Seiten gesetzt werden, denn die $\text{\LaTeX}$ -Ausgabe erfolgt unter Verwendung der Dokumentklasse book.

Kapitel 1

Die Philosophie von L^AT_EX

„Onkel Cosmo, warum nennt man das Ding Textverarbeitung?“

„Ganz einfach Skyler: Du hast doch gesehen was die Lebensmittelverarbeitung mit dem Essen anstellt, oder?“

– Jeff MacNelly in "Shoe"

1.1 Was ist L^AT_EX

L^AT_EX ist ein komplettes System zur Erstellung von Dokumenten. Es ist unübertroffen beim Schreiben von komplexen wissenschaftlichen und technischen Artikeln mit mathematischen Formeln, Querverweisen, Literaturlisten, Indexlisten usw. Es kann sehr gut mit Texten jeglicher Größe umgehen, bei denen die üblichen Anforderungen gestellt werden: Automatische Seiten- und Kapitelnumerierung, Rechtschreibprüfung usw. Natürlich kann man L^AT_EX auch verwenden um einen Brief an die Mutter zu schreiben, obwohl es dafür sicherlich einfachere Programme gibt. Es ist sicherlich nicht das geeignetste Werkzeug um damit Werbezettel, Flugblätter oder ähnliches zu erstellen (warum werden wir etwas später erläutern), obwohl auch das mit einigem Aufwand sehr wohl möglich ist. Hier einige Beispiele wozu L^AT_EX im allgemeinen verwendet wird: Memos, Briefe, Dissertationen und Diplomarbeiten, Bücher (über PostgreSQL, Fernabastung, Kryptologie, Novellen, Poesie und sogar ein oder zwei Kinderbücher), Artikel in referierten wissenschaftlichen Zeitschriften, Skripte für Theaterstücke und Filme, wirtschaftliche Proposals...

Abbildung 14.14: Auszug aus einem L^AT_EX-Dokument

Deutlich zu erkennen ist der unterschiedliche Zeilenumbruch, der in der Oberfläche an die aktuelle Breite des X11-Fensters angepasst ist. Seitenumbrüche fehlen in der Oberfläche ganz. Die Position des Cursors ist durch den dünnen senkrechten Strich am Anfang der Überschrift sichtbar. Das Zeichen  am Ende der beiden ersten Zeilen der Überschrift steht für explizite Zeilenumbrüche an diesen Stellen. Kaum zu erkennen ist hingegen, dass L^AT_EX für die Bildschirmausgabe einen PostScript-Zeichensatz statt der T_EX-Zeichensätze verwendet. Der Benutzer kann natürlich den eingestellten Bildschirm-Font (Zeichensatz) ändern.

Die Arbeit mit L^AT_EX ist analog zu anderen vom PC bekannten Textverarbeitungssystemen (Benutzung der Maus, Pulldown- und Popup-Menüs, Icons); aufgrund dieser Ähnlichkeiten dürften Benutzer, die bereits mit anderen Textverarbeitungssystemen gearbeitet haben, nur geringe Schwierigkeiten haben, sich in der Bedienung von L^AT_EX zurechtzufinden. Die Icons zum Laden eines Dokuments, zum Drucken sowie zum Ausschneiden (Cut), Kopieren (Copy) und Einfügen (Paste) von Teilen des Dokuments in der Aktionsleiste (Toolbar) unterscheiden sich nicht wesentlich von denen anderer Textverarbeitungsprogramme. Wird die Maus auf ein Icon bewegt, erscheint nach kurzer Zeit die gelb hinterlegte Be-

zeichnung der zugehörigen Aktion, etwa ‚Ausschneiden‘ beim Icon zum Ausschneiden von Teilen des Dokuments.

Links von den genannten Icons befindet sich ein Feld, das anzeigt, welches Textelement an der Cursorposition bearbeitet wird. Da der Cursor in Abbildung 14.13 am Anfang des Dokument-Titels steht, wird aktuell ‚Titel‘ angezeigt. Durch Anklicken des rechts daneben befindlichen Icons ► wird ein Pulldown-Menü geöffnet, das alle vorgegebenen Textelemente anzeigt. Durch Auswahl eines anderen als des gerade aktiven Textelements kann das aktuelle Textelement oder ein mit der Maus selektierter Teil des Dokuments in ein anderes Textelement umgewandelt werden. Normaler Text entspricht dem Textelement ‚Standard‘. Andere Textelemente sind beispielsweise Überschriften oder mathematische Formeln.

Rechts von den bisher beschriebenen Icons finden sich in der Aktionsleiste eine Reihe weiterer Symbole, die nicht geläufig sind. Sie werden daher kurz beschrieben:

-  Hervorheben von Text durch Verwenden des Slanted-Fonts (`\emph`) statt der Normalschrift;
-  Hervorheben des Texts durch Verwenden der Kapitälchen (`\textsc`), gedacht vor allem zum Auszeichnen von Eigennamen;
-  Öffnen eines Menüs zum Einstellen des zu verwendenden Fonts;
-  Modus zur Eingabe von L^AT_EX-Befehlen;
-  Einfügen einer Formel im Fließtext. Der Satz erfolgt dann in der von T_EX gewohnten hohen Qualität.
-  Erstellen von Fußnoten (`\footnote`);
-  Erstellen von Randnotizen (`\marginpar`);
-  Verschieben von Teilen von Listen in äußere oder innere Listen;
-  Einbinden von Bildern;
-  Erstellen von Tabellen.

14.3.1.1 Das Menü Datei

Am oberen Rand des L^AX-Fensters sind die aktuell verfügbaren Pull-down-Menüs angeordnet. Neben den üblichen Einträgen enthält das Menü Datei die Einträge **Neu von Vorlage...** zum Laden vorgefertigter Dokumentformen und **Wieder herstellen** zum Restaurieren des letzten gesicherten Zustands. Erwähnenswert ist der Eintrag **Öffnen...**, der ein Fenster zur

Auswahl der zu ladenden Datei öffnet. Bemerkenswert ist dabei, dass dieses Fenster einen ‚Button‘ **Beispiele** hat, über den etliche als Teil der Distribution mitgelieferte Beispiele in mehreren Sprachen geladen werden können.

Über das Untermenü **Versionskontrolle** können mehrere Versionen eines Dokuments verwaltet und zugänglich gemacht werden (mit Hilfe von RCS, ‚Revision Control Program‘). Weitere Untermenüs erlauben das Importieren und Exportieren in verschiedenen Formaten. Importieren lassen sich neben $\text{\LaTeX}$ - auch ASCII-, LinuxDoc- sogar Word-Dateien. Ein Export ist als ASCII-, .dvi-, HTML-, $\text{\LaTeX}$ -, PDF- oder PostScript-Datei möglich.

14.3.1.2 Das Menü Bearbeiten

Im Menü **Bearbeiten** finden sich neben **Rückgängig** und **Wiederholen** auch die üblichen ‚Cut and Paste‘-Operationen und Einträge zum ‚Suchen und Ersetzen‘. Über ein Untermenü können externe Textpassagen als Absätze oder zeilenweise eingefügt werden. Der Aufruf eines Rechtschreib-Prüfprogramms (ispell) ist ebenfalls in diesem Menü angeordnet. Die beim Auftreten von Fehlern beim Formatieren mit $\text{\LaTeX}$ eingesetzten ‚Fußnoten‘ können über einen weiteren Eintrag komplett entfernt werden. Darüberhinaus enthält das Menü die folgenden Einträge:

Tabelle

Dieses Untermenü fasst Einträge zusammen, mit denen das Layout einer Tabelle bearbeitet werden kann (Spalten, Abgrenzungen durch Linien, spaltenübergreifende Tabelleneinträge)

Mathe

Einträge in diesem Untermenü erlauben das Ein- und Ausschalten der Formelnummerierung und den Austausch des Formeltyps. Analog zum vorhergehenden Menü finden sich hier die Einträge zum Einstellen des Layouts eines array.

Eine Besonderheit ist das Untermenü **Computer-Algebra-System verwenden**. In Formelumgebungen ist der Zugriff auf Octave, Maxima, Mathematica und mehrerer Varianten von Maple vorgesehen.

TeX prüfen

Aufruf des Programms `chktex` (nicht Teil der Distribution) zum Überprüfen des Syntax der $\text{\LaTeX}$ -Datei.

Gleitobjekte öffnen/Schließen

Schalter zum Wechsel zwischen der Wiedergabe von Abbildungen, Tafeln und Einfügungen wie Fußnoten oder Randnotizen und ihrer symbolischen Darstellung.

Einstellungen...

Dieser Eintrag öffnet ein Fenster, in dem die Einstellungen für LyX vorgenommen werden.

Neu konfigurieren

Über diesen Eintrag lässt sich LyX nach Änderungen an von LyX verwendeten Programmen oder der $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ -Installation neu konfigurieren.

14.3.1.3 Das Menü Einfügen

Das Pulldown-Menü **Einfügen** bietet Möglichkeiten zum Einbauen spezieller Textelemente. Das erste Untermenü **Mathe** dient dem Einfügen der verschiedenen Formelumgebungen von $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ und aus dem Paket `amsmath`. Über **Mathe** → **Mathe-Kontrollfläche** lässt sich ein Fenster öffnen, in dem man Zugriff auf die im mathematischen Modus verwendeten Operatoren, Relationen, Funktionen und Symbole hat. Es schließt sich das Untermenü **Sonderzeichen** an, über das Hoch- und Tiefstellen möglich ist oder spezielle Zwischenräume wie `\hfill`, das doppelte Anführungszeichen oder Fortsetzungspunkte, ... eingefügt werden können.

Zitat... öffnet im Fall eines mit Hilfe von $\text{BIB}_{\text{T}}\text{E}_{\text{X}}$ aufgebauten Literaturverzeichnisses ein Fenster, in dem die Schlüssel der Einträge in der `.bib`-Datei als Liste angezeigt werden. Bei Auswahl eines Schlüssels werden die zugehörigen Daten angezeigt. Es besteht sogar die Möglichkeit, die Daten der Einträge zu suchen; bei Erfolg wird der zugehörige Schlüssel hervorgehoben und die Daten werden angezeigt. Aus dieser Liste kann ein Schlüssel übernommen werden in die Liste der an dieser Stelle gewünschten Literaturverweise – bei $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ die Argumente des `\cite`-Befehls. Das Literaturverzeichnis selbst wird über das Untermenü **Listen und Inhaltsverz.** weiter unten aufgebaut.

Marken lassen sich über den Eintrag **Marke...** und Querverweise über **Querverweis...** einfügen. **Querverweis...** öffnet dabei ein Fenster, in dem alle definierten Marken angezeigt werden und in mehreren Formatvarianten übernommen werden können. Über **Fußnote** und **Randnotiz** können Fußnoten und Randnotizen erstellt werden. Einträge in das Stichwortverzeichnis können über **Stichwort...** vorgenommen werden. Der Eintrag **Notiz** erlaubt das Einfügen von Notizen, die in der Druckfassung nicht erscheinen.

Listen und Inhaltsverz. öffnet ein Untermenü, über das an der Cursorposition das Inhaltsverzeichnis, das Verzeichnis der Abbildungen oder Tabellen, das Stichwortverzeichnis, das Literaturverzeichnis oder ein Verzeichnis der Algorithmen (eine Variante des Abbildungs- oder Tabellenverzeichnisses) eingefügt werden kann; die genannten Verzeichnisse wer-

den innerhalb von LyX allerdings nur durch eine Markierung angezeigt und nicht aufgebaut.

Der Eintrag **TeX** erlaubt die Eingabe von $\text{L}\text{A}\text{T}\text{E}\text{X}$ -Code. **Minipage** fügt eine `minipage`-Umgebung ein. Klickt man mit der rechten Maustaste in den Bereich der Minipage, öffnet sich ein Fenster, in dem die Breite und Ausrichtung der Minipage festgelegt werden kann. Die Einträge **Grafik...** und **Tabelle...** entsprechen den Icons in der Aktionsleiste (Toolbar) und fügen ein Bild oder eine Tabelle ein. $\gg\text{Floats}\ll$, also Abbildungen, Tafeln und Algorithmen, können über das Untermenü **Gleitobjekte** eingefügt werden. Die folgenden Einträge fügen ASCII- oder LyX -Dateien ein. Der Eintrag **Externes Material...** ermöglicht beispielsweise das Einbinden von Diagrammen oder XFig-Ausgaben.

14.3.1.4 Das Menü Format

Im Menü **Format** sind Funktionen zusammengefasst, die das Layout des gesamten Dokuments oder der aktuellen Textstelle festlegen.

- Der Eintrag **Zeichen...** öffnet ein Fenster, in dem sich Font und Farbe für die Zeichen einstellen lassen.
- In dem mit **Absatz...** geöffneten Fenster kann man Eigenschaften von Abschnitten festlegen, beispielsweise die Einrückung am ersten Zeilenanfang und die Zeilenabstände.
- **Dokument...** öffnet ein Fenster, in dem die Grundeinstellungen für das Dokument vorgenommen werden können, darunter die Dokumentklasse und ihre Optionen, der Seitenstil, die Sprache des Dokuments (wichtig für Trennungen und automatisch eingefügte Bezeichnungen wie „Inhaltsverzeichnis“ oder „Table of Contents“) und der Font für das Dokument (Auswahl zwischen TEX - und einigen PS-Schriften). Das Papierformat, die Orientierung der Seite und die Seitenränder werden in diesem Fenster eingestellt. Auch die in `itemize`-Listen zu verwendenden Markierungssymbole können hier festgelegt werden. Die Präambel für das Formatieren des Dokuments mit $\text{L}\text{A}\text{T}\text{E}\text{X}$ kann ebenfalls in diesem Fenster ergänzt werden (beispielsweise durch Laden zusätzlicher Pakete oder Festlegen eigener Befehle); von LyX automatisch eingefügte Teile der Präambel werden dabei nicht angezeigt. Die aktuellen Layout-Einstellungen können über den ‚Button‘ **Als Dokument-Standards speichern** als Voreinstellungen für künftige Arbeiten mit LyX übernommen werden.
- Über die Einträge **Hervorhebung**, **Kapitälchen** und **Fettdruck** können unterschiedliche Hervorhebungen des Texts bei der Eingabe aktiviert werden.

- Über die Einträge **Umgebungstiefe verringern** und **Umgebungstiefe erhöhen** können Teile von Listen in äußere oder innere Listen verschoben werden.
- Den Übergang vom Hauptteil eines Dokuments zu den Anhängen ermöglicht der Eintrag **Anhang hier beginnen**.

14.3.1.5 Das Menü Ansicht

Hauptaufgabe des Menüs ist es, das Dokument zu formatieren und in verschiedenen Formaten in einem eigenen Fenster anzuzeigen. Unterstützt wird neben dem `.dvi`-Format HTML, PDF und PostScript. Über das Untermenü **Aktualisieren** können nach Änderungen am Dokument neue Versionen für die Anzeige generiert werden. Daneben enthält das Menü auch die folgenden Einträge.

LaTeX-Protokoll

Zeigt die `.log`-Datei des letzten $\text{\LaTeX}$ -Aufrufs an (mit Möglichkeit zum Aktualisieren)

Inhaltsverzeichnis

Anzeigen des Inhaltsverzeichnisses in einem eigenen Fenster, in dem man Überschriften anklicken kann, um zu diesen Abschnitten zu springen.

TeX-Informationen

Öffnet ein Fenster, in dem die in $\text{\LaTeX}$ umgesetzten $\text{\LaTeX}$ -Klassen, -Pakete und $\text{\BIBTeX}$ -Stile angezeigt werden. Über den ‚Button‘ **Neu lesen** können die Listen aktualisiert werden. **Ansicht** öffnet ein weiteres Fenster, in dem die $\text{\LaTeX}$ -Umsetzung angezeigt wird.

14.3.1.6 Das Menü Navigieren

Dieses Menü bildet die Dokumentstruktur ab, ähnlich wie das Inhaltsverzeichnis, und ermöglicht so den schnellen Wechsel zu anderen Kapiteln und Abschnitten. Daneben können über Einträge in diesem Menü auch die folgenden speziellen Objekte angesprungen werden:

Fehler

Sprung an die Stelle des nächsten Fehlers beim Formatieren des Dokuments.

Notiz

Sprung zur nächsten $\text{\LaTeX}$ -internen Notiz (erscheint nicht im Ausdruck).

Marke

Sprung zur nächsten Marke im Dokument.

Den Abschluss bildet das Untermenü **Lesezeichen**, in dem bis zu drei Lesezeichen gesetzt und angesprungen werden können.

14.3.1.7 Die Menüs Dokumente und Hilfe

Über das Pulldown-Menü **Dokumente** kann man zu jedem der geladenen Dokumente wechseln. Der Name der aktuell bearbeiteten Datei wird am unteren Rand des **LyX**-Fensters angezeigt.

Hilfe bietet neben der in Abbildung 14.13 gezeigten ‚Einführung in **LyX**‘ eine Reihe weiterer Dokumentationen zu **LyX** als Online-Hilfe an. Auch die Informationen zum Programm und den Entwicklern können abgerufen werden.

14.3.2 T_EXmacs

Die WYSIWYG-ähnliche Oberfläche **T_EXmacs** wurde von Joris van der Hoeven als frei verfügbarer wissenschaftlicher Texteditor entwickelt. **T_EX** und GNU Emacs beeinflussten das Design. Ziel war eine benutzerfreundliche Oberfläche mit der Möglichkeit, strukturierte Dokumente zu schreiben. Da auch automatisch generierte mathematische Formeln mit der hohen Satzqualität umgesetzt werden, eignet sich **T_EXmacs** auch als Interface zu Computeralgebra-Systemen.

Die Menüs und die Online-Hilfe stehen in mehreren Sprachen – darunter neben Englisch auch Deutsch und Französisch – zur Verfügung. Der Wechsel zwischen den Sprachen ist über das unten beschriebene Icon  (*Eine Sprache auswählen*) leicht möglich.

Standardmäßig werden Dokumente in einem internen Format mit der Erweiterung **.tm** des Dateinamens gespeichert. Neben der Ausgabe als **L^AT_EX**-Datei ist auch ein Export als HTML-, SCHEME- oder XML-Datei möglich. Fließtext kann auch als Textdatei ohne Formatierung exportiert werden. Umgekehrt können solche Dateien auch importiert werden. Vorgesehen ist zusätzlich der Export als PDF- oder als PostScript-Datei. Für die Zukunft ist eine Weiterentwicklung zu einer vollständigen wissenschaftlichen Office Suite geplant.

Das Aussehen von **T_EXmacs** bei der Arbeit mit einem ‚normalen‘ Text zeigt Abbildung 14.15. Die Bedienung des **T_EXmacs** wird anhand dieses Bilds beschrieben. Die untere Icon-Leiste und das Pulldown-Menü **Text**

werden kontextabhängig ausgetauscht, etwa beim Bearbeiten mathematischer Formeln oder – wie Abbildung 14.16 zeigt – während einer Sitzung. Auch solche Varianten werden kurz erläutert.

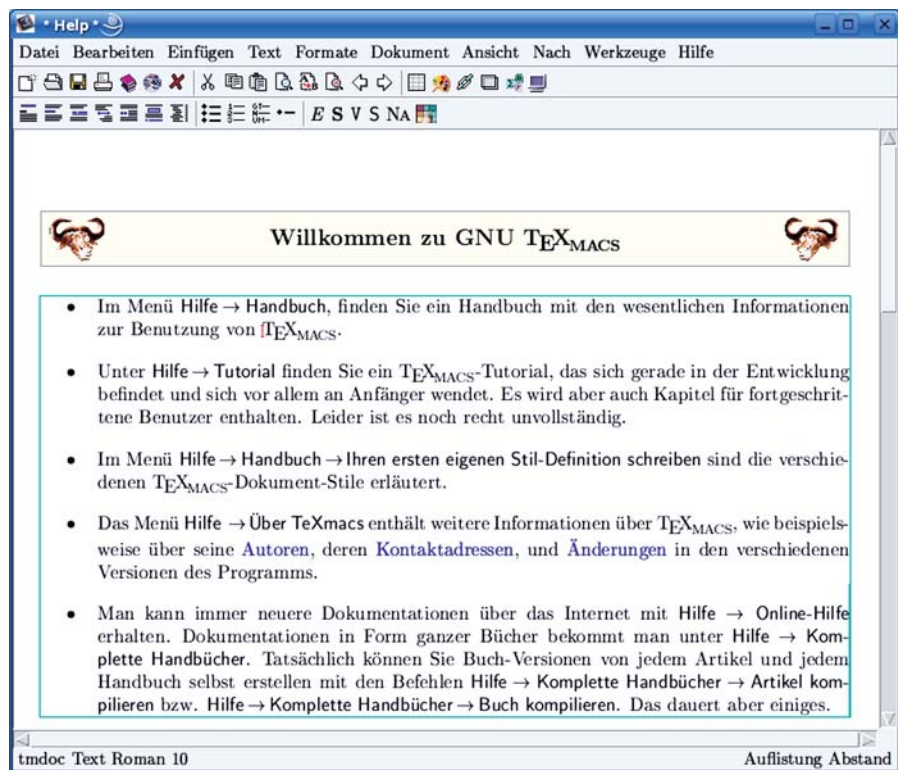


Abbildung 14.15: Begrüßungsdatei des TeXmacs

14.3.2.1 Pulldown-Menüs

Zunächst werden die Pulldown-Menüs behandelt. Der Aufruf kann auch über die rechte Maustaste erfolgen, welcher ein Popup-Menü mit diesen Menüs zugeordnet ist. Ganz links in der Kopfleiste findet sich das Menü *Datei*. Dieses hat neben den üblichen Einträgen zum Laden und Speichern von Dateien und zum Beenden des TeXmacs auch Untermenüs für eine Reihe von Voreinstellungen. Hierüber kann man ferner Dateien Exportieren und Importieren aus anderen Formaten – darunter auch L^AT_EX.

Im daran anschließenden Menü **Bearbeiten** findet man die gewohnten Funktionen zum Zurücknehmen/Wiederausführen von Änderungen, zum Kopieren/Ausschneiden markierter Teile des Dokuments und zum Einfügen zwischengespeicherter Passagen. Neben dem üblichen (unbenannten) Zwischenspeicher stehen im $\text{T}_{\text{E}}\text{X}$ macs weitere Zwischenspeicher zur Verfügung, die für diese Operationen genutzt werden können. Der Zugriff erfolgt über Untermenüs in diesem Menü. Auch die Rechtschreibprüfung und das Untermenü für die Einstellungen des $\text{T}_{\text{E}}\text{X}$ macs sind hier zu finden.

Das Pulldown-Menü **Einfügen** besteht aus einer Sammlung von Untermenüs, die verschiedene Bereiche abdecken. So gibt es etwa Untermenüs für die Befehle im Zusammenhang mit Querverweisen, zum Einfügen von Bildern und Tabellen sowie für Befehle im Zusammenhang mit dem Zeilen- und Seitenumbruch. Andere Untermenüs enthalten seitenbezogene Funktionen für Kopf- und Fußzeilen und Seitennummern. Schließlich gibt es auch einige Untermenüs für spezifische Funktionen des $\text{T}_{\text{E}}\text{X}$ macs. Das Untermenü **Mathematik** bietet unterschiedliche mathematische Umgebungen an. Wählt man eine dieser Umgebungen aus, wechselt $\text{T}_{\text{E}}\text{X}$ macs in den mathematischen Modus. Über das Untermenü **Sitzung** kann man eine der später erläuterten Sitzungen öffnen.

Das nächste Pulldown-Menü wird kontextabhängig ausgetauscht. Normalerweise findet sich an dieser Stelle das Menü **Text**, über das eine Vielzahl von in Untermenüs organisierten $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ -Umgebungen und -Befehlen abgerufen werden kann. Ein weiteres Untermenü erlaubt das Erstellen verschiedener Verzeichnisse und Register.

Beim Bearbeiten von Tabellen wird an dieser Stelle das zusätzliche Pulldown-Menü **Tabelle** eingeblendet. Über eine Reihe von Untermenüs kann eine Vielzahl von Eigenschaften der Tabelle festgelegt werden. Einstellungen – darunter Umrandungen, Hintergrundfarben oder die Ausrichtung des Texts – können wahlweise für einzelne Zellen, Zeilen oder Spalten oder für die gesamte Tabelle vorgenommen werden. Selbst Untertabellen werden in diesem Menü angeboten.

Im mathematischen Modus des $\text{T}_{\text{E}}\text{X}$ macs tritt das Menü **Mathematik** an die Stelle des Menüs **Text**. Gleichzeitig wird die untere Icon-Leiste ausgetauscht, um den Zugriff auf die mathematischen Zeichen und Funktionen zu erleichtern. Das Menü erlaubt den Zugriff auf die wichtigsten mathematischen Funktionen und die – auf mehrere Untermenüs verteilten – mathematischen Sonderzeichen.

Auch beim Öffnen einer Sitzung wird das Menü **Text** ausgetauscht, und zwar gegen das Pulldown-Menü **Sitzung**. Auch in diesem Fall wird die untere Icon-Leiste angepasst. Dieses Menü enthält eine Reihe von Aktionen,

die bei Sitzungen hilfreich sind, etwa zum Einfügen von Anmerkungen in den Ablauf der Ausgabe einer Sitzung.

Das Pulldown-Menü **Formate** gehört wieder zu den permanent angezeigten Menüs. Die erste – stets vorhandene – Gruppe von Untermenüs dient der Festlegung der Schriftattribute und -größe. Untermenüs der folgenden Gruppe erlauben die Wahl der Schriftfarbe und der aktuellen Sprache. Im mathematischen Modus steht an dieser Stelle eine Gruppe von Untermenüs zur Festlegung stilistischer Eigenschaften im mathematischen Modus. Den Abschluss bildet eine Gruppe von Untermenüs, über die Absätze ausgerichtet, Ränder und Abstände eingestellt und die Anzahl der Spalten (bis zu drei) definiert werden können. Zusätzlich ist hier ein Untermenü zu finden, in dem man zwischen normaler und professioneller Trennung wählen kann.

Im Pulldown-Menü **Dokument** sind Einstellungen und Festlegungen für das gesamte Dokument zusammengefasst. Hier werden der Dokumentstil und die zu verwendenden Pakete festgelegt. Auch andere globale Einstellungen können über dieses Menü vorgenommen werden. Nicht unwichtig dürfte das Untermenü **Aktualisieren** sein, über das einzelne oder alle automatisch erstellten Verzeichnisse aktualisiert werden können.

Das anschließende Menü **Ansicht** fasst Funktionen zusammen, mit denen sich das Aussehen des $\text{\TeX}$ macs beeinflussen lässt. Hier kann man etwa festlegen, welche Kopfzeilen im Fenster angezeigt werden sollen. Über Einträge in diesem Menü kann man auch weitere Fenster öffnen, um gleichzeitig mehrere Einzeldokumente zu bearbeiten.

Arbeitet man mit mehreren Puffern, so kann man über das Menü **Nach** leicht zwischen diesen Puffern wechseln. Sobald ein neuer Puffer – also ein neues Dokument – geöffnet wird, wird es um einen entsprechenden Eintrag ergänzt.

Über das Menü **Werkzeuge** ist es möglich, Systembefehle oder SCHEME-Programme auszuführen. Hier wird auch eingestellt, welcher Typ von Dateien im- oder exportiert werden soll. Ein weiteres Untermenü erlaubt einige Aktualisierungen. Schlussendlich können über dieses Menü auch Webseiten erstellt werden. Bei Dokumenten, die aus mehreren Teildokumenten bestehen, kann im Untermenü **Projekt** ein Hauptdokument festgelegt („Attach master...“) oder entfernt werden („Detach master“).

Im Pulldown-Menü **Hilfe** ist eine ausführliche Online-Dokumentation zu finden. Der Übersichtlichkeit halber ist sie auf mehrere Untermenüs verteilt. Über eines der Untermenüs lässt sich auch eine komplette Benutzeranleitung (über 100 Seiten) erzeugen, die auch ausgedruckt werden kann. Die Dokumentation liegt in verschiedenen Sprachen – darunter

auch Deutsch – vor. Die aktuell eingestellte Sprache legt zugleich fest, in welcher Sprache die Dokumentation angezeigt wird.

14.3.2.2 Icons

Unterhalb der Leiste mit den Pulldown-Menüs sind zwei Zeilen mit Icons angeordnet – falls sie nicht über das Menü *Ansicht* ausgeblendet wurden. Die obere Zeile wird in der Regel stets beibehalten, während die untere im mathematischen Modus ausgetauscht, bei der Bearbeitung von Tabellen erweitert und bei Sitzungen angepasst wird.

Die Icons in der oberen Zeile sind in drei Gruppen aufgeteilt: Die erste dieser Gruppen enthält Icons zum Laden und Speichern von Dateien, zum Drucken, für einige besonders wichtige Einstellungen und zum Schließen von Dateien sowie zum Beenden des Programms. In der zweiten Gruppe sind die wichtigen Editierfunktionen zu finden. Die dritte Gruppe bilden Icons zum Einfügen von Tabellen, Abbildungen und Querverweisen. Das nächste Icon erlaubt den Zugriff auf Funktionen des $\text{\TeX}$ macs im Zusammenhang mit Präsentationen. Ein weiteres Icon dient zum Einbinden und Verwalten von Animationen und Sound. Besonders wichtig ist das daran anschließende Icon zum Einfügen mathematischer Formeln. Den Abschluss dieser Gruppe bildet ein Icon, über das interaktive Verbindungen zu anderen Programmen wie gnuplot oder einem Kommandointerpreter aufgebaut werden können; die Eingaben und Ergebnisse werden in das Dokument übernommen.

In der folgenden Beschreibung wird jeweils auch die Kurzbeschreibung in der ersten Zeile wiedergegeben, wie sie beim Berühren des Icons mit der Maus erscheint. Ist einem Icon ein Menü zugeordnet, erscheint in der ersten Zeile zusätzlich ein entsprechender Hinweis.



Neuen Puffer öffnen

(Menü)

Über dieses Menü kann man einen neuen Puffer oder ein neues Fenster öffnen, um ein anderes Dokument zu bearbeiten. Es besteht auch die Möglichkeit, das aktuelle Fenster zu ‚klonen‘, also ein neues Fenster für das gleiche Dokument aufzumachen.



Datei laden

(Menü)

Hier kann man ein Dokument in einen Puffer laden oder auf den gespeicherten Zustand zurücksetzen. Es ist auch möglich, ein Dokument in einem neuen Fenster zu öffnen. Zusätzlich kann man über dieses Menü Dateien in Fremdformaten – darunter $\text{\LaTeX}$ -Dokumente – importieren.

*Puffer sichern*

(Menü)

Dieses Menü dient zum Speichern des aktuellen Puffers, wobei der Dateiname vorgegeben werden kann (oder muss, wenn das Dokument neu angelegt und noch nicht gespeichert wurde). Außerdem kann der Puffer in mehreren Fremdformaten, darunter $\text{\LaTeX}$ und HTML) gespeichert werden. Auch ein Export als PostScript- oder PDF-Datei ist möglich.

*Drucken*

(Menü)

Neben einer Druckvorschau mit Ghostview wird hier das Drucken des Puffers auf einem Drucker oder in eine Datei angeboten, wobei die Möglichkeit besteht, die zu druckenden Seiten auszuwählen.

*Dokumentstil auswählen*

(Menü)

In diesem Menü kann man den gewünschten Dokumentstil – in $\text{\LaTeX}$ -Sprechweise eine Dokumentklasse – aus einer Reihe von vorgegebenen Stilen auswählen.

*Sprache auswählen*

(Menü)

Hier kann die Sprache für die Oberfläche und das Dokument ausgewählt werden. Neben den im Dokument automatisch generierten Bezeichnungen werden beispielsweise auch die Bezeichnungen der Pulldown-Menüs ausgetauscht.

*Schließen*

(Menü)

Die Einträge in diesem Menü erlauben das Schließen des aktuellen Puffers, des aktuellen Fensters oder das Beenden des $\text{\TeX}$ macs.

Die zweite Icon-Gruppe besteht aus den üblichen Editierfunktionen. Da die Kurzbeschreibungen selbsterklärend sind, wird auf eine weitere Beschreibung verzichtet.

*Text ausschneiden* ([C-w])*Text kopieren* ([M-w])*Text einfügen* ([C-y])*Text suchen* ([C-s])*Ersetzen mit Rückfragen* ([C-=])*Text auf Schreibfehler überprüfen* ($\text{[M-\$]}$)

 *Letzte Änderungen stornieren* (-)

 *Stornierte Änderungen wiederherstellen* (-)

Beim Betrachten der Online-Hilfe werden anstelle der letzten beiden Icons zum Stornieren und Wiederherstellen von Änderungen die beiden folgenden Icons angezeigt, mit denen man zum vorherigen oder folgenden Thema der Online-Hilfe wechseln kann.

 *Browse back*

 *Browse forward*

Die dritte Gruppe von Icons hilft beim Einfügen von Tabellen, Abbildungen und Querverweisen. Hier sind auch der Zugriff auf Präsentationen, Animationen und Sound angesiedelt. Am wichtigsten dürfte das Icon für mathematische Formeln sein. Das letzte Icon erlaubt den Zugriff auf interaktive Verbindungen. Den Icons sind Menüs zugeordnet, die hier beschrieben werden.

 *Tabelle einfügen* (Menü)

Dieses Icon erlaubt den direkten Zugriff auf das Untermenü **Einfügen** → **Tabelle**. Die beiden ersten Einträge in diesem Menü fügen eine Tabelle ($\LaTeX$ -Umgebung `table`) in das Dokument ein, die sich nur in der Größe unterscheidet. Abweichend von den $\LaTeX$ -Konventionen handelt es sich bei diesen Tabellen allerdings nicht um Gleitobjekte; bei einem Export nach $\LaTeX$ wird vielmehr statt einer `table`-Umgebung eine `minipage` verwendet, die auch die Tabellenunterschrift enthält.

Die folgenden vier Einträge legen tabelliertes Material analog der $\LaTeX$ -Umgebung `tabular` an, die anfänglich nur eine Spalte und eine Zeile haben. Im Fall der ‚Block‘-Einträge sind die einzelnen Zellen mit einem Rahmen versehen, der beim ‚Tabulatormodus‘ fehlt. Zusätzlich hat man beim Anlegen die Wahl zwischen linksbündigen und zentrierten Spalten. Nach Auswahl eines dieser Einträge befindet man sich im Tabellenmodus.

Im Tabellenmodus können in einfacher Weise Zeilen und Spalten hinzugefügt werden. Die Eigenschaften einzelner Zellen, ganzer Zeilen bzw. Spalten oder auch der gesamten Tabelle lassen sich leicht anpassen. Die Ausrichtung der Zellinhalte, Strichstärke und Art der Umrandungslinien oder die Abstände der Zellinhalte vom Rand der Zelle sind modifizierbar. Über eines der tabellenspezifischen Icons kann man eine Untertabelle in eine Zelle einfügen. Auch Zellen, die sich über mehrere Spalten und Zeilen erstrecken, werden unterstützt.

**Bild einfügen**

(Menü)

Wie bei Tabellen kann man über die beiden ersten Einträge große oder kleine Abbildungen (Umgebung `figure`) einfügen. Hier handelt es sich aber nicht um Gleitobjekte im Sinne von $\LaTeX$. Der Eintrag **Draw image** baut ein ein Grafikobjekt auf, das mit einem integrierten Grafikeditor erstellt und bearbeitet werden kann. Die beiden folgenden Einträge binden vorhandene `.gif`-, `.jpeg`-, `.pdf`-, `.png`-, `.pnm`-, `.ppm`-, `.svg`-, `.tiff`-, `.xpm`-, `Xfig`- oder PostScript-Bilder ein. Bei **Bild verknüpfen...** wird wie bei $\LaTeX$ nur der Dateiname mit einigen zusätzlichen Informationen in der Dokumentdatei abgelegt. Im Fall von **Bild einbinden...** wird die komplette Bilddatei in die Dokumentdatei übernommen. Über **Thumbnails...** können Thumbnails eingebunden werden. Den Abschluss bildet das Untermenü **Plot**, über das Kurven (zweidimensional) und Oberflächen (dreidimensional) aufgebaut werden können. Auch parametrische Kurven und Oberflächen werden unterstützt.

**Beziehung einfügen**

(Menü)

Die ersten drei Einträge in diesem Menü ermöglichen Querverweise und sind die $\TeX$ macs-Varianten der $\LaTeX$ -Befehle `\label`, `\ref` und `\pageref`. Der folgende Eintrag **Einfügen...** erlaubt das Übernehmen von Dateien in Schreibmaschinenschrift und entspricht damit dem $\LaTeX$ -Befehl `\verbatimfile`; er ist besonders geeignet, um (Programm-)Listings im Dokument wiederzugeben. Der Eintrag **Hyperlink** fügt einen Link auf eine WWW-Seite oder eine lokale Datei ein. Über den Eintrag **Aktion** kann ähnlich wie bei einem Hyperlink einem Text ein GUILF- oder SCHEME-Script zugeordnet werden, das durch einen Doppelklick auf den Text ausgeführt werden kann.

Bei den drei letzten Einträgen handelt es sich um Menüs. **Zitat** enthält Einträge, die den $\LaTeX$ -Befehlen `\cite` und `\nocite` entsprechen. Der Untereintrag **Mit Info** entspricht dem `\cite`-Befehl mit optionalem Argument. Unter **Indexeintrag** findet man – wie nicht anders zu erwarten – die Befehle zum Anlegen bis zu dreistufiger Indexeinträge. Das letzte Untermenü **Glossareintrag** stellt die Befehle zum Aufbau eines Glossars zur Verfügung.

**Zwischen Textebenen schalten**

(Menü)

Dieses Menü enthält Einträge, um Präsentationen zu gestalten. Im wesentlichen werden zwei Effekte zur Verfügung gestellt. Der eine Effekt besteht darin, zwischen verschiedenen Varianten einer Textpassage umzuschalten. Ein zweiter Effekt erlaubt es, Textpassagen ein- bzw. auszublenden (zu falten). Über das Menü können die Effekte in das Dokument eingefügt und gesteuert werden.

*Animation* (Menü)

Über Einträge in diesem Menü ist es möglich, animierte GIF-Dateien (Filme) und Sounddateien in Dokumente einzubauen und zugehörige Einstellungen vorzunehmen.

*Mathematik einfügen* (Menü)

Das Menü bietet die $\text{\LaTeX}$ -Formelumgebungen an.

*Eine interaktive Sitzung beginnen* (Menü)

Die Einträge in diesem Menü stellen interaktive Verbindungen zu anderen Programmen wie gnuplot oder einer Shell her. Ein- und Ausgaben werden protokolliert und in das Dokument übernommen.

In der zweiten Zeile sind kontextabhängige Icons angeordnet. Im Textmodus besteht sie beispielsweise aus drei Gruppen. Die Icons der ersten Gruppe erlauben den Zugriff auf Menüs zur Struktur des Dokuments. Die zweite Gruppe enthält Menüs, über die unterschiedliche Listen in das Dokument eingefügt werden können. Durch die Icons der letzten Gruppe kann die Schriftart ausgewählt und über ein Menü die Farbe der Schrift eingestellt werden.

*Einen Titel eingeben* (Menü)*Einen neuen Abschnitt beginnen* (Menü)*Eine Umgebung einfügen* (Menü)*Absatzformatierung* (Menü)*Absatzränder einstellen* (Menü)*Fußnoten und bewegliche Objekte einfügen* (Menü)*Automatisch erzeugte Verzeichnisse einfügen* (Menü)*Auflistung* (Menü)*Aufzählung* (Menü)*Beschreibende Auflistung* (Menü)*Text hervorheben (F5)*

S *Kräftigen Text schreiben (F6)***V** *Wörtlichen Text schreiben (F7)***S** *Beispieltext schreiben (F8)***NA** *Einen Namen schreiben (S-F6)**Eine Vordergrundfarbe auswählen*

(Menü)

Beim Bearbeiten von Tabellen (in der $\text{\LaTeX}$ -Umgebung `tabular`) kommen zwei Gruppen mit Tabellenfunktionen hinzu. Über die Menüs der ersten dieser Gruppen können Eigenschaften der gesamten Tabelle festgelegt und Zeilen oder Spalten hinzugefügt und gelöscht werden. In die zweite Gruppe wurden die Menüs eingeordnet, über die sich Eigenschaften einer Zelle oder einer Gruppe von Zellen einstellen lassen. Das erste Menü dieser Gruppe dient daher zum Festlegen, ob eine einzelne Zelle, eine Spalte, eine Zeile oder die gesamte Tabelle verändert werden soll. Darüberhinaus kann man mit der Maus auch einen rechteckigen Zellenblock auswählen. Das zweite Menü dient zum Ausrichten der Zellinhalte, das dritte für Einstellungen im Zusammenhang mit Höhe und Breite der Tabelle, das vierte zum Einstellen von Rändern und Umrandungen, das fünfte zum Festlegen der Hintergrundfarbe und das letzte für spezielle Einstellungen.

Im mathematischen Modus besteht die zweite Zeile hingegen aus drei Gruppen von Icons. Die erste Gruppe enthält mathematische Funktionen wie Brüche, Wurzeln, Hoch- und Tiefstellungen und ein Menü mit den mathematischen Akzenten. In den Menüs der zweiten Gruppe sind die mathematischen Operatoren und Symbole zu finden. Die dritte Gruppe entspricht der vierten (und ebenfalls letzten) Gruppe im Textmodus und dient der Auswahl des zu verwendenden mathematischen Fonts und der Schriftfarbe.

Eine Spezialität des $\text{\TeX}$ macs ist die Möglichkeit, im Rahmen einer ‚Sitzung‘ mit externen Systemen ähnlich wie mit einer Shell zu kommunizieren. Das Untermenü **Einfügen** $\rightarrow$ **Sitzung** bietet vordefinierte Sitzungen unter anderem für Scheme (Eintrag **Schema**), GNUplot (Eintrag **Gnuplot**), Graphviz (Eintrag **Graphviz**) und eine Kommando-Shell (Eintrag **Shell**) an. Über den Eintrag **Anders...** können weitere Sitzungstypen aufgerufen werden, für die jedoch ein Interface definiert sein muss. Insbesondere für einige Computeralgebra-Systeme wird ein solches Interface mitgeliefert. Auch die Definition eigener Interfaces ist möglich und wird in der

Dokumentation beschrieben. Abbildung 14.16 zeigt den Umgang mit einer Sitzung am Beispiel von GNUplot: Ein- und Ausgabe – einschließlich grafischer Ausgaben – werden in das Dokument übernommen. Die Abbildung zeigt eine aktive Sitzung (zu erkennen am Rahmen um den Bereich der Sitzung und das in der Abbildung geöffnete Pulldown-Menü **Sitzung** anstelle von **Text**). Nach Abschluss der Sitzung über den Eintrag **Sitzung** → **Sitzung schließen** bleibt der Inhalt der Sitzung (Ein- und Ausgabe) erhalten. Damit kann man $\text{\TeX}$ macs auch als Oberfläche für diese Systeme einsetzen und darüber hinaus das Arbeiten mit diesen Systemen dokumentieren.

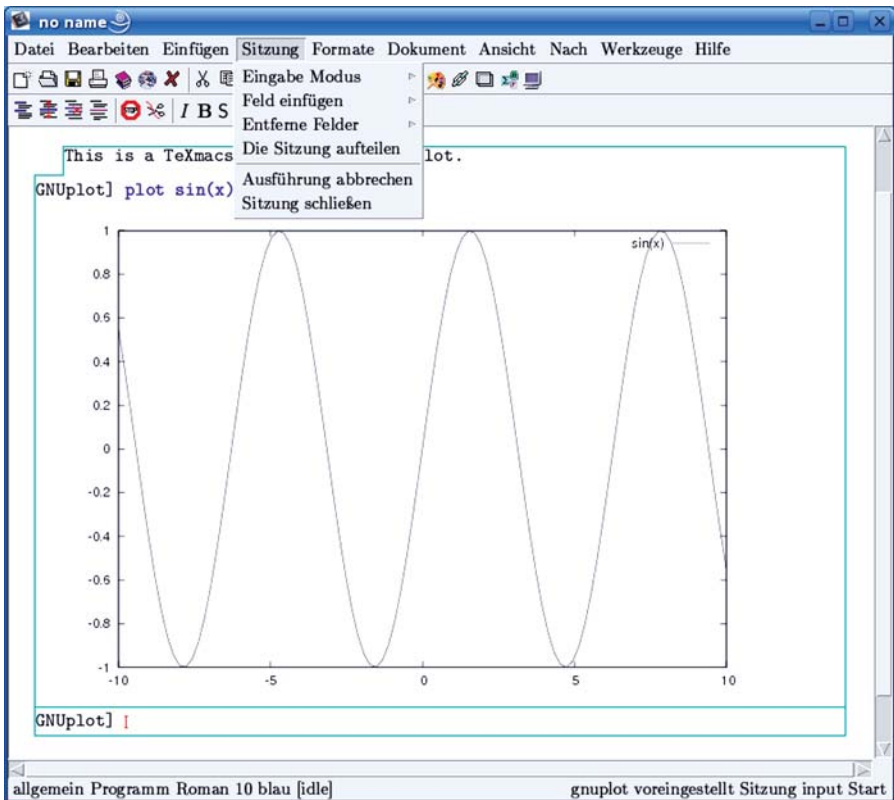


Abbildung 14.16: Eine aktive GNUplot-Sitzung im $\text{\TeX}$ macs

Feinheiten der Typografie¹

15.1 Einleitung

Jeder, der sich mit $\LaTeX$ und $\TeX$ beschäftigt, ist auch mehr oder weniger mit Typografie befasst. Man kann die Typografie grob in zwei Gebiete aufteilen: Makrotypografie und Mikrotypografie. Die Makrotypografie umfasst Dinge wie Satzspiegel und Seitenlayout, Schriftarten und -grade, Titel, Register, usw. Zur Mikrotypografie zählen zum Beispiel der richtige Gebrauch von Anführungszeichen, der Abstand einer Zahl von ihrer Einheit und die Verwendung von Strichen verschiedener Länge. Sie befasst sich also mit dem Satz von Buchstaben, Symbolen und Wörtern.

Die makrotypografischen Regeln liegen bei sehr vielen wissenschaftlichen Publikationen fest. Seien sie für Bücher und Zeitschriften vom Verlag vorgegeben, teilweise sogar in verlagseigenen $\LaTeX$ -Klassen-Dateien vordefiniert oder in Prüfungsordnungen für Diplom- und Doktorarbeiten festgeschrieben. Auch wenn dies nicht der Fall ist, wird eine Autorin/ein Autor in aller Regel auf ein von anderen erstelltes Layout wie z. B. das KOMA-Script-Paket zurückgreifen, denn die Erstellung eines eigenen Buchlayouts ist ein Prozess, der sehr viel Vorbildung, Kreativität und Zeit benötigt.

Bei der Anwendung der Mikrotypografie geht es darum, einem Text, der inhaltlich „steht“ und nach einem vorgegebenen oder auch selbstgestalteten Layout gesetzt wurde, den letzten Schliff zu geben. Natürlich kann man gewisse Regeln schon bei der Eingabe berücksichtigen, jedoch sind beispielsweise Trennungen ein Problem, welches man möglichst erst im Endstadium des Publikationsprozesses angeht, da man dieses sonst

¹ Dieser Text wurde zuerst veröffentlicht in *Die $\TeX$ nische Komödie*, Mitgliederzeitschrift von DANTE, Deutschsprachige Anwendervereinigung $\TeX$, e. V., Heft 4/96 und 1/97.

mehrfach bedenken muss. Eine Abwägung von Aufwand versus Nutzen bei der vollständigen Anwendung aller Mikrotypografie-Regeln ist durchaus sinnvoll, jedoch sollte der Qualitätsanspruch bei einem wissenschaftlichen Buch sehr hoch angesetzt werden. Ein Buch wird von unterschiedlichen Lesern zu unterschiedlichen Zeiten und nicht unbedingt linear gelesen. Die Einhaltung der Mikrotypografie-Regeln bietet bei den verschiedenen Leseansprüchen Hilfestellungen. Bei der Erstellung eines Buchs ist der Inhalt die Hauptsache. Er hat höchste Priorität. Anders gesagt: „Der Geist kommt vor der Form“. Jedoch zeichnet sich eine exzellente Publikation eben auch durch die Eleganz der Darstellung aus.

Die Regeln der Mikrotypografie stellen ein klar abgegrenztes Gebiet dar, welches jedoch ziemlich verstreut in diversen Büchern beschrieben ist. Manche Angaben stehen in direktem Widerspruch zueinander, oder es werden zu einer generellen Regel eine Reihe von Sonderfällen genannt, und eine weitere Anwendungsregel besagt, dass die generelle Regel ohne Ausnahme gilt. Dieser Artikel versucht eine Übersicht der geltenden Traditionen zu geben, genauer gesagt ist er eine Zusammenstellung der in der genannten Literatur gefundenen Regeln. Die Formulierungen und Beispiele sind zum Teil wörtlich übernommen ohne dies als Zitat zu kennzeichnen, da der Text dann zu unübersichtlich wäre.

Natürlich sind ein Teil der nachfolgend beschriebenen Regeln willkürliche Festlegungen. Man kann über Zweckmäßigkeit oder Unzweckmäßigkeit einzelner Definitionen sicher heftig streiten. Jeder, der sich an bestimmte Gestaltungsregeln gewöhnt hat, wird gute Gründe dafür haben und ungern davon abgehen. Auch hat die psychologische Forschung gezeigt, dass es einem durchschnittlichen Leser nach kurzer Übungszeit gelingt, Texte mit extrem abweichendem Aussehen flüssig zu lesen, beispielsweise Texte ohne jeglichen Wortzwischenraum (siehe auch Gross (1994)). Dennoch haben die nachfolgend erläuterten Mikrotypografie-Regeln ihren Sinn, zu einem kleinen Teil in der Lesbarkeit, zu einem viel größeren Teil in der Ästhetik von Texten. Sie dienen ferner der Förderung und Verbesserung der schriftlichen Kommunikation, denn Regeln beugen Missverständnissen beim Verstehen vor.

Keine Regel ist so feststehend, dass sie nicht mit guten Gründen gebrochen werden kann. Bei verschiedenen Verlagen werden unterschiedliche Regeln angewandt, teilweise sogar individuell pro Buch. Innerhalb eines Buchs jedoch müssen sie *einheitlich* sein.

Es werden die Regeln für den deutschsprachigen Raum beschrieben. so weit möglich und sinnvoll werden auch fremdsprachliche Regelungen aufgeführt. In Zweifelsfällen oder bei widersprüchlichen Regeln wurde der Duden (Dudenredaktion, 1996) als Standardwerk bevorzugt.

15.2 Abkürzungen, Akronyme, Einheiten, Himmelsrichtungen

Abkürzungen sollten soweit wie möglich vermieden werden – selten steht der erzielte Platzgewinn in sinnvoller Relation zur Lesbarkeit des abgekürzten Textes.

Gebräuchliche Abkürzungen sind in (Dudenredaktion, 1996, S. 14 f.) sowie [6] aufgeführt; ferner in [9], für juristische Werke in [10] und für die EDV [8]. Eigene Abkürzungen sollten sprechend sein.

Ein Akronym ist ein aus den Anfangsbuchstaben mehrerer Wörter gebildetes Wort, z. B. Aids, DANTE, GAL. Die Schreibweise kann man in [6] nachschlagen; im Zweifelsfall ist ein Akronym in Großbuchstaben zu setzen. Für Akronyme gilt im Wesentlichen dasselbe wie für Abkürzungen.

Abkürzungen sind dann einzusetzen, wenn

- a) die Abkürzung üblich und der Leser damit vertrauter ist als mit der Langform oder

Beispiel: z. B., Dr., usw.

- b) der Platzgewinn beträchtlich ist und schwerfällige Wiederholungen vermieden werden.

Alle Abkürzungen, die nicht unter die Kategorie a) fallen, sind, auch wenn sie noch so vertraut erscheinen, vor ihrer ersten Verwendung im Text einzuführen. Dies geschieht entweder mittels eines Abkürzungsverzeichnisses, oder der abzukürzende Ausdruck wird beim ersten Auftreten im vollen Wortlaut angegeben, unmittelbar gefolgt von der in Klammern eingeschlossenen und später verwendeten Abkürzung.

Beispiel: Elektroenzephalogramm (EEG)

Grundsätzlich müssen alle Abkürzungen im gesamten Werk *einheitlich* sein. Bei allen Einheiten sind die neuesten DIN (Deutsche Industrienormen) bzw. SI-Normen (Système International d'Unités) anzuwenden.

Beispiel: m² statt qm, J statt cal

15.2.1 Schreibweise von Abkürzungen

- Ein Satz sollte nicht mit einer Abkürzung beginnen, gegebenenfalls verwendet man die Langform; ausgenommen von dieser Regel sind Fußnoten.
- Bei Abkürzungen wird die Beugungsendung meist nicht wiedergegeben.

Beispiel: des Jh., des Bd., des PC

Bei Pluralbildung dagegen wird die Beugungsendung häufig angegeben, um Missverständnissen besonders bei weiblichen Abkürzungen vorzubeugen, da hier der Artikel im Singular und Plural gleich lautet.

Beispiel: die GmbHs; auch: die Bde.

Gelegentlich wird der Plural durch Buchstabenverdopplung ausgedrückt.

Beispiel: Jgg., ff.

- Abkürzungen, die im vollen Wortlaut gesprochen werden, und fremdsprachliche Abkürzungen erfordern einen Punkt.

Beispiel: z. B., usw., Dr.; k. o., ib.

Dies gilt auch für einige Abkürzungen, die besonders in der Alltagssprache nicht mehr ausgesprochen werden.

Beispiel: a. D., i. V.

- Nach Abkürzungen, die als selbständige Wörter gesprochen werden, folgt kein Punkt.

Beispiel: IQ, EEG, NATO, BGB; Ausnahme: F.D.P., manchmal auch U.S.A.

Ebensowenig nach vom Autor selbst definierten (und eingeführten, siehe oben) Abkürzungen.

- Einzelne Teile mehrgliedriger Abkürzungen werden durch einen kleinen Zwischenraum (Spatium) getrennt.

Beispiel: z. B.

Eingabe: z . \ , B .

In englischsprachigen Texten wird zwischen den Abkürzungsteilen kein Zwischenraum gesetzt.

- Bei Maßbezeichnungen sind in Verbindung mit Ziffern die Einheiten zu verwenden; ohne Ziffern werden sie ausgeschrieben.

Beispiel: 10 kg/cm, aber: wenige Millisekunden

Abgekürzte Währungs- und Maßeinheiten sowie die Zeichen f./ff. und die dazugehörige Zahl sollten nicht durch einen Zeilenumbruch voneinander getrennt werden. Dazu muss als Eingabe statt eines Leerzeichens eine ~ geschrieben werden.

Beispiel: 14 ff., 10 €

Eingabe: 14~ff., 10~\texteuro

In englischsprachigen Texten wird zwischen Zahl und Einheit nur ein kleiner Zwischenraum gesetzt. Die Eingabe \, statt eines Leerzeichens erzeugt einen entsprechenden Zwischenraum.

Auf Abkürzungen der Maße, Gewichte, Himmelsrichtungen, der meisten Währungseinheiten, der chemischen Elemente und der Einheiten in den Naturwissenschaften folgt kein Punkt.

Beispiel: m, kg, SW, F (für Franc), €, NaCl

Ausnahmen sind Abkürzungen von Zahlwörtern, bei diesen wird ein Punkt gesetzt.

Beispiel: Tsd., Mio., Mrd.

- Steht eine Abkürzung mit Punkt am Satzende, dann ist der Abkürzungspunkt zugleich Schlusspunkt des Satzes.

15.2.2 Liste gebräuchlicher Abkürzungen

(Bei mehreren möglichen Schreibweisen ist die kürzeste zu verwenden.)

a. a. O.	am angegebenen Ort (nicht aaO)	ad lib.	zu dem Buch
ca.	circa	bzw.	beziehungsweise (nicht bezw.)
dgl.	dergleichen	cf.	confer! (= vergleiche)
d. h.	das heißt	dsgl.	desgleichen
evtl.	eventuell	etc.	et cetera (= und so weiter)
ff.	fortfolgende	f.	folgende
i. Allg.	im Allgemeinen	ggf.	gegebenenfalls
i. E.	im einzelnen	i. d. F.	in der Fassung (nicht idF)
resp.	respektive	m. E.	meines Erachtens
s.	siehe	Rn.	Randnummer
sog.	sogenannt(e)	S.	Seite
u. a.	unter anderem	St.	Stück (nicht Stck.)
usw.	und so weiter	u. Ä.	und Ähnliches
u. U.	unter Umständen	u. E.	unseres Erachtens
v. a.	vor allem	u. v. a. m.	und vieles andere mehr
z. B.	zum Beispiel	vgl.	vergleiche (nicht vergl.)
z. T.	zum Teil	z. H.	zu Händen (nicht z. Hd.)
		z. Z.	zur Zeit (nicht z. Zt.)

Abkürzungen/Akronyme

- möglichst vermeiden
- einführen: im Text oder Abkürzungsverzeichnis
- DIN, SI-Normen beachten
- nicht am Satzbeginn, außer bei Fußnoten

Beugungsendung des Jh., des Bd., des PC

Pluralbildung die GmbHs, die Bde., Jgg., ff.

Im Wortlaut gespr. z. B., d. h., usw., Dr.; z. \, B.
 aber: a. D., i. V. a. \, D.

In Buchstaben gespr. IQ, EEG, NATO/Nato; aber: F.D.P.

Bei Einheiten 10 kg/cm 10~kg/cm
 € 5,- €~5,-

15.3 Anführungszeichen, Zitate

Anführungszeichen, im Deutschen auch Gänsefüßchen bzw. Hochkommas genannt, stehen vor und hinter

- einer wörtlich wiedergegebenen Äußerung (direkte Rede),
- einer wörtlich angeführten Textstelle (Zitat),
- zitierten Überschriften, Titeln von Büchern, Filmen, Gedichten, Namen von Zeitungen und Ähnlichem,

Beispiel: „Die Zeit“

- Wortschöpfungen und Worten, die im übertragenen Sinne gemeint sind (Metaphern); jedoch wirken bei bekannten Metaphern Anführungszeichen eher übertrieben,

Beispiel: Das Bier ‚kostete‘ ihn den Führerschein.

- einzelnen Wortteilen, Wörtern oder Textteilen, die hervorgehoben werden sollen.

Bei Zitaten sind folgende allgemeine Regeln zu beachten: Zitate sind immer wörtlich wiederzugeben, sie dürfen weder im Wortlaut noch in Rechtschreibung und Interpunktion vom Original abweichen. Eigene Korrekturen oder Ergänzungen im Zitat sind durch eckige Klammern, Auslassungen (Ellipsen) durch Auslassungspunkte gegebenenfalls in eckigen Klammern zu kennzeichnen.

Beispiel: „Je länger, je lieber‘ oder ‚kurz und knapp‘? Wenn er [der Autor] zum Imponieren neigt, ...“

Längere Auszüge aus urheberrechtlich geschützten Werken, wobei unter „länger“ circa eine halbe Textseite zu verstehen ist, dürfen nicht ohne Genehmigung des Autors bzw. Verlags zitiert werden. Für Lehre und Forschung gibt es Ausnahmen, die in § 51 UHRG geregelt sind.

Lange Zitate werden nicht in Anführungszeichen eingeschlossen, sondern eingerückt mit einer `quote-` oder `quotation-`Umgebung gesetzt.

15.3.1 Schreibweisen von Anführungszeichen und halben Anführungszeichen

- Deutsche Anführungszeichen sind „...“

Beispiel: „Ja, bitte!“

Eingabe: "``Ja, bitte!``" oder
`\glqq{}Ja, bitte!\grqq{}`

(Die Namen `glqq` und `grqq` sind dabei die Abkürzungen für „german left/right quote“.)

und als sogenannte spitze Form »...«.

Beispiel: »Ja, bitte!«

Eingabe: ">Ja, bitte!"<

Sie werden ohne Zwischenraum direkt an den anzuführenden Text gesetzt.

- Eine Anführung innerhalb einer Anführung wird durch halbe Anführungszeichen deutlich gemacht.

Beispiel: „Viele verwenden den Begriff ‚Sozialethik‘, ohne sich darunter etwas vorstellen zu können.“

Eingabe: " `Viele ...
 \glq{}Sozialethik\grq{} , ...
 k"onnen. " "

(Die Namen `glq` und `grq` sind dabei die Abkürzungen für „german left/right quote“.)

- Stehen Anführungszeichen und halbe Anführungszeichen direkt beieinander, so werden sie durch einen kleinen Zwischenraum (Spatium) getrennt. Dieser wird vom Paket `ngerman` bzw. durch entsprechende Unterschneidungen (Kerning) der verwendeten Schrift automatisch eingefügt.

Beispiel: „Sing doch nicht immer ‚La Paloma““

Eingabe: " `Sing ... immer \glq{}La
 Paloma\grq""

- In englischsprachigen Texten sind die Anführungszeichen “...” und ‘...’.

Beispiel: “quoted text”,
 ‘single-quoted text’

Eingabe: ``quoted text``,
 `single-quoted text`

In Frankreich und der Schweiz sind als Anführungszeichen «...» üblich, wobei im Französischen beide Zeichen von Spatien umschlossen werden.

Beispiel: « Première citation »

Eingabe: "<\,Premi\`ere
 citation\,"> oder \flqq\,Pre-
 mi\`ere citation\,\frqq{}

Bei Verwendung des Pakets `babel` mit der Option `french` können die französischen Anführungszeichen auch einfacher eingegeben werden

Beispiel: « Première citation »

Eingabe: \og{}Premi\`ere
 citation\fg{}

Eine Anführung innerhalb einer Anführung wird durch halbe Anführungszeichen deutlich gemacht. Die halbe Form «...» wird in französischen Texten nicht verwendet.

Beispiel: «Liest du gerade ‹Homo faber›?»

Eingabe: "<Liest du gerade
\flq{}Homo faber\frq?">

(Die Namen `flq` und `frq` stehen für „french left/right quote“.)

- Die Eingabe des Zeichens " (auf der deutschen Tastatur über der Ziffer 2) als Anführungszeichen ist falsch.
- Es gibt ein spezielles Zeichen für die Maßeinheiten Zoll und Bogensekunde.

Beispiel: 3,5''

Eingabe: 3,5\(^{\prime}\prime\)

- Bei einem Zitat aus einer Fremdsprache werden entweder die deutschen Anführungszeichen oder die `quote-` bzw. `quotation-` Umgebungen gewählt. Nur wenn innerhalb des Zitats fremdsprachliche Anführungszeichen auftreten, sind diese zu übernehmen.
- Bei allen Schriften mit OT1-Kodierung, darunter die CM-Schriften, fehlt das Kerning zwischen Anführungszeichen und nachfolgendem bzw. vorhergehendem Zeichen. Bei einigen Buchstaben-Anführungszeichen-Kombinationen können daher zu große oder zu kleine Abstände auftreten.

Beispiel: „V statt „V

Der Einsatz von T1-kodierten Schriften behebt das Problem.

Anführungszeichen

Deutsch, 1. Form	„ “	" \ " ' oder \glqq \grqq{}
Deutsch, 2. Form	» «	"> "<
Deutsch, halbe	, ‘	\glq \grq{}
Englischsprachig	“ ”	\ \ ’ ’
Englischspr., halbe	‘ ’	\ ’
Französisch	« »	"<\, \, ">, \flqq\, \, \frqq{}
Schweizerisch	« »	"< "> oder \flqq \frqq{}
Schweizerisch, halbe	< >	\flq \frq{}
Zoll, Bogensekunden	"	\(^{\prime}\prime\)

15.4 Satzzeichen, Auslassungs- und Fortsetzungspunkte

Die Regeln zur Verwendung der Satzzeichen wie Punkt, Komma, Semikolon, Doppelpunkt, Ausrufungszeichen und Fragezeichen werden als allgemein bekannt vorausgesetzt.

15.4.1 Schreibweise von Satzzeichen

- Vor einem Satzzeichen steht niemals ein Leerzeichen, immer jedoch dahinter.

Im Französischen wird ein kleiner Zwischenraum vor und nach den Satzzeichen ; , : , ! und ? eingefügt.

- Auslassungspunkte, Ellipsenzeichen genannt, kennzeichnen den Abbruch einer Rede, das Verschweigen eines Gedankenabschlusses oder Auslassungen in Zitaten.

Beispiel: Der Autor sagt: „Ja, aber ...“

Die Auslassungspunkte werden von Wortzwischenraum umgeben, wenn sie für ein selbständiges Wort oder mehrere Wörter stehen. Bei Auslassung eines Wortteils werden sie unmittelbar an den Rest des Wortes angeschlossen. Ein Zeilenumbruch vor den Auslassungspunkten ist zu vermeiden. Dazu wird in der Eingabe ~ statt eines Leerzeichens verwendet.

Beispiel: Ja, aber ...

Eingabe: Ja, aber~\dots{ }

Stehen die Auslassungspunkte am Satzende, wird kein zusätzlicher Schlusspunkt gesetzt; noch deutlicher: im Anschluss an Auslassungspunkte wird niemals ein Punkt gesetzt. Sonstige Satzzeichen und Anführungszeichen folgen ohne Zwischenraum.

15.5 Verschiedene Sonderzeichen, E-Mail-Adressen

- Das Et-Zeichen & ist gleichbedeutend mit „u.“ (und), darf aber laut [5] nur bei Firmenbezeichnungen (Personen-Firmen) verwendet werden.

Beispiel: Voß & Co.

Eingabe: Vo"s \& Co.

Oft wird bei der Angabe von Literatur, die in Klammern im Text steht, zwischen Autoren das Et-Zeichen benutzt.

Beispiel: (Schulze, Müller & Meier, 1993)

- Bei Temperaturangaben ist zwischen der Zahl und dem Gradzeichen ein Wortzwischenraum zu setzen; der Kennbuchstabe der Temperaturskala folgt ohne weiteren Zwischenraum.

Beispiel: 17 °C

Eingabe: 17~\textcelsius

Bei anderen Gradangaben wird das Gradzeichen ohne Zwischenraum an die Zahl angeschlossen.

Beispiel: ein Winkel von 30°

Eingabe: ... von 30\textdegree{}

- Steht das Wort „Paragraph“ in Verbindung mit einer nachgestellten Zahl, dann setzt man das Zeichen §, sonst wird das Wort ausgeschrieben.

Beispiel: § 9

Eingabe: \S~9

Zwei Paragraphenzeichen (§§) kennzeichnen den Plural.

Beispiel: §§ 10–15

Eingabe: \S\S~10--15

Mit den CM-Schriften ist der Abstand zwischen zwei Paragraphenzeichen zu groß, man muss sie enger zusammenrücken. Ein Makro hierfür könnte beispielsweise lauten:

```
\newcommand{\SSS}{\S\S\kern-.4pt\S\relax}
```

Beispiel: §§ 10–15

Eingabe: \SSS~10--15

Anmerkung: § wird von T_EX, obwohl es ein Textzeichen ist, intern als Symbol in einer mathematischen Formel gesetzt, da dieses Zeichen in den CM-Text-Schriften nicht vorhanden ist. Verwendet man nun andere Textschriften, wie beispielsweise PostScript-Schriften, müssen über NFSS passende mathematische Schriften verwendet werden.

- Vor Prozent- und Promillezeichen ist ein kleiner Zwischenraum (Spatium) zu setzen.

Beispiel: 25 %

Eingabe: 25\,\%

Dieser Zwischenraum entfällt bei Ableitungen oder Zusammensetzungen.

Beispiel: eine 25%ige Steigerung, die 5%-Hürde

In englischsprachigen Texten wird kein Zwischenraum gesetzt.

Anmerkung: Das Promillezeichen ist in OT1-kodierten Schriften, speziell den CM-Schriften, nicht vorhanden. Bei Verwendung des Pakets `textcomp` und kann es mit der Anweisung `\textperthousand` gesetzt werden. Bei der Schriftfamilie `wasy` steht es mit den Befehlen `\usepackage{wasysym}` und `\permil` zur Verfügung. Für PostScript-Schriften gilt dieselbe Anmerkung wie für das Paragraphenzeichen „§“.

- Die Symbole für Warenzeichen (*trademark*) und eingetragenes Warenzeichen (*registered trademark*) werden ohne Zwischenraum direkt an das Wort gesetzt, das gesetzlich geschützt ist. Alternativ kann auch der Text „Warenzeichen“ bzw. „eingetragenes Warenzeichen“ vor oder in Klammern nach dem Warenzeichen stehen.

Beispiel: $\mathrm{T\!E\!X}^{\mathrm{TM}}$, $\mathrm{T\!E\!X}$ ist ein
Warenzeichen der American
Mathematical Society®

Eingabe: `\TeX\texttrademark,`
`\TeX{} ist ...`
`Society\textregistered`

Das Zeichen für den Urheberrechtsschutz wird durch einen Wortzwischenraum von Jahr und Inhaber des Rechts getrennt, alternativ wird der Begriff „Copyright“ ausgeschrieben.

Beispiel: © 1996 Springer-Verlag

Eingabe: `\copyright~1996`
`Springer-Verlag`

- Fremdsprachige Namen werden mit allen Sonderzeichen wie im Original geschrieben.

Beispiel: Ångström

Eingabe: `\AA{}ngstr"om`

- Bei der Angabe von E-Mail- und WWW-Adressen muss man heutzutage keine besonderen Vorkehrungen oder Schriftauszeichnungen vornehmen.

Ein Zeilenumbruch sollte bevorzugt nach dem Zeichen @ und sonst nur nach den Zeichen !, \$, %, – und . ohne Einfügung eines Trennstrichs erfolgen.

Beispiel: Hugo.Tintenfisch@
KorallenRiff.Meer

Eingabe: `Hugo."Tintenfisch@"`
`Korallen"=Riff."Meer`

Beispiel: `http://lammarsch.uni-hd.de/lbsv`

Eingabe: `http://lammarsch""uni
"=hd.""de/""lbsv`

Innerhalb der Teilstücke können, da "" nur zusätzliche Trennstellen einfügt und keine Trennungen verhindert, weitere Trennungen vorkommen. Abhilfe schafft das Paket `path` von Philip Taylor, das ausschließlich die genannten erlaubten Trennstellen setzt.

Eingabe: `\path|Hugo.Tintenfisch@Korallen-Riff.Meer|`

Sonderzeichen

Fortsetzungspunkte	also ...	<code>also~\dots</code>
	al...	<code>al\dots</code>
Et-Zeichen	&	<code>\&</code>
Grad	30°C	<code>30\textcelsius{}</code>
Paragraph	§ 9	<code>\S~9</code>
	§§ 10–15	<code>\SSS~10-15, \S\S~10-15</code>
Prozent	25 %	<code>25\, \%</code>
Warenzeichen	TM	<code>\texttrademark{}</code>
Eingetr. Warenzeichen	®	<code>\textregistered{}</code>
Copyright	©	<code>\copyright{}</code> , <code>\textcopyright{}</code>

15.6 Ligaturen

Ligaturen sind Zusammenfassungen mehrerer Buchstaben, die im Wortstamm zusammengehören. Historisch sind z. B. ß und & Ligaturen. Der Buchstabe „ß“ ist nach Angaben in [3, Stichwort „Eszett“] etwa im 14. Jahrhundert im Fraktursatz aus der Ligatur von f (s) und z (z) zu dem Buchstaben ß verschmolzen. Die Transliteration als sz ist jedoch falsch, richtig ist ss. Eine andere Theorie besagt, dass das Eszett aus den beiden verschiedenen s-Buchstaben, dem Mittel-s f und dem Ende-s-Zeichen s, entstanden ist. Eine gute Literaturstelle zu diesem Thema ist [11].

Die Herkunft von Ligaturen erklärt sich aus den Handschriften. Bei Gutenberg wurde der Randausgleich durch den Einsatz von Buchstabenkombinationen an Stelle von Einzelbuchstaben erzielt, nicht wie heute durch Variation des Wortabstands.

Gebräuchliche Ligaturen sind: ff, fi und fl. $\TeX$ verwendet diese Ligaturen automatisch, falls sie in der Schrift existieren.

Beispiel: Pfiff, erfinden, Pfandflasche

15.6.1 Schreibweise von Ligaturen

- Zur Aufhebung von Ligaturen bietet das Paket `ngerman` den Befehl `"|`, der einen kleinen Zwischenraum einfügt und zusätzlich diese Stelle als Trennstelle markiert. (Die Eingabe erfolgt über die Taste links unten mit den Zeichen `>/<.`)

Beispiel: Auflage

Eingabe: Auf"|lage

- Keine Ligatur steht zwischen Wortstamm und Endung, außer bei der Ligatur „fi“.

Beispiel: schaufle, höflich;
aber: streifig

Eingabe: schauf"|le, h"of"|lich

- Keine Ligatur steht in der Wortfuge von Zusammensetzungen, außer beim Wegfall des dritten Konsonanten f.

Beispiel: Auflage, Schaffell,
Schiffahrt

Eingabe: Schaf"|fell,
Schiff"|fahrt

(Vgl. auch den Abschnitt über Worttrennungen, Teil II.)

- Schließt eine Abkürzung mit zwei Buchstaben, die eine Ligatur bilden können, dann wird diese angewendet.

Beispiel: Aufl.

Eingabe: Aufl.

- Im Zweifelsfall werden die Ligaturen nach Sprechsilben aufgehoben.

Beispiel: knifflige Sauerstoffflasche

Eingabe: kniff"|lige
Sauerstoff"|flasche

- Ligaturen des Fremdsprachensatzes werden auch gesetzt, wenn fremdsprachige Wörter vereinzelt in deutschen Texten vorkommen:

- Die Ligaturen in englischsprachigen Wörtern sind dieselben wie im Deutschen. Zusätzlich gibt es noch: ffi und ffl.

Beispiel: fluffier

- In französischen Wörtern werden zusätzlich die Ligaturen Œ und œ verwendet.

Beispiel: Œuvres

Eingabe: `\OE{}uvres`

- Im Dänischen, Norwegischen und Isländischen gibt es die Ligaturen Æ und æ.

Beispiel: Ærø

Eingabe: `\AE{}r\o{}`

- Bei lateinischen Wörtern dürfen nur Ae, ae, Oe, oe gebraucht werden, nicht die französischen und dänischen Ligaturen.

Beispiel: Asa foetida

- Im polnischen und tschechischen wird ck als Ligatur gesetzt mit Ausnahme von Eigennamen.

Beispiel: Drucksache, aber: Hrdlicka, Hrdlicka

Eingabe: `\textfrac{Drucksache},`
`\textfrac{Hrdlic" |ka}`

(In den CM-Schriften ist keine ck-Ligatur vorhanden.)

- Im Niederländischen sind die Zeichen i und j als Ligatur zu setzen.
- Im Spanischen wird Ll und ll als Ligatur gesetzt.

15.7 Zahlen und Ziffern

Generelle Regel: Zahlen ab 10 werden durch Ziffern, Zahlen kleiner als 10 durch Worte dargestellt.

Beispiel: fünf Schritte, 43 Schritte

Ein Satz oder eine Überschrift sollte nicht mit einer Zahl beginnen; gegebenenfalls ist die ausgeschriebene Form zu verwenden, noch besser ist eine Umformulierung des Satzes (vor allem bei Zahlen größer 10).

Ausnahmen: Zahlen kleiner als 10 werden als Ziffern dargestellt, wenn

- sie in einem Vergleich mit Zahlen größer als 10 im selben Satz stehen,

Beispiel: 3 von 35 Versuchspersonen, Der Umsatz stieg um 3,5 %, der Gewinn um 2 %; im Gegensatz zu: Sie haben fünf Antwortmöglichkeiten

- ihnen eine Einheit voraus geht oder folgt oder sie eine bestimmte Stelle in einer nummerierten Folge darstellen, wobei die Zahlen „Null“ und „Eins“ trotzdem als Worte gesetzt werden, wenn sie dadurch leichter zu verstehen sind.

Beispiel: € 5,-, 2 mg, Meßwert 4, Kap. 3, 1 Stunde, 5 %, Tür Nummer 6;
 aber: das mit der Ziffer Null versehene Feld, eine von drei Personen

15.7.1 Schreibweise von Zahlen

- Ganze Zahlen aus mehr als 3 Ziffern werden von der Endziffer aus in dreistellige Gruppen zerlegt. Gegliedert wird durch einen kleinen Zwischenraum (Spatium), nicht durch Komma. Eine Gliederung durch Punkt ist möglich, kann aber zu Verwechslungen führen, da in englischsprachigen Texten der Punkt die Dezimalstelle angibt.

Beispiel: 314 159

Eingabe: 314\,159

Dezimalstellen werden laut [7] von den ganzen Zahlen durch Komma getrennt. Auch nach dem Komma wird, vom Komma ausgehend, in Dreiergruppen gegliedert.

Beispiel: 2,718 281 8

Eingabe: 2,718\,281\,8

Diese Regel wird in sehr vielen Büchern verschiedener Verlage zugunsten der internationalen Schreibweise mit Punkt abgeändert. Dies war übrigens bis zum 18. Jahrhundert auch eine in Deutschland übliche Schreibweise, neben etlichen anderen, eingeführt 1492 von Pellos, die jedoch – möglicherweise unter französischem Einfluß – vom Dezimalkomma abgelöst wurde (vgl. [1]).

Beispiel: 2.718 281 8

Bei Zahlen kleiner als Eins werden führende Nullen angeführt, außer wenn die Zahl selbst nicht größer als Eins werden kann, beispielsweise bei Wahrscheinlichkeiten oder Korrelationen.

Beispiel: 0,25; aber: $p = ,315$, $P = .315$

- Zwischen einem Zahlenwert und einer vorhergehenden oder nachfolgenden Einheit darf kein Zeilenumbruch erfolgen; ebenso zwischen einem Zahlenwert und einer Abkürzung. Dazu wird in der Eingabe ~ statt eines Leerzeichens verwendet.

Beispiel: 5 km, 5 f., S. 5, Bd. 5

Eingabe: 5~km, 5~f., S.~5, Bd.~5

(Vgl. dazu auch den Abschnitt „Mathematischer Satz“, S. 569.)

- Folgt einer Zahl ein einzelner Buchstabe, der Teil einer Nummerierung ist, so wird in der Regel ein kleiner Zwischenraum gesetzt.

Beispiel: Tab. 12 a, b, Fig. 20 a–c,
§ 11 a BGB

Eingabe: Tab.~12\, a, \, b,
Fig.~20\, a--c, \S~11\, a BGB

- Bei der Angabe von Geldbeträgen in Euro werden die Cent durch ein Komma abgetrennt. Der Betrag und die Einheit € bzw. ¢ dürfen nicht getrennt werden.

Beispiel: 10 000 €, € 25,–

Eingabe: 10\, 000~\texteuro,
\texteuro~25,--

Beispiel: ¢ 25, 10 ¢

Eingabe: 10~\textcent,
\textcent~25

- Bei Angabe eines Datums wird zwischen die einzelnen Datumsteile ein Punkt gesetzt.

Beispiel: 27.3.1996

Wenn möglich wird der Monat in Langform angegeben. Dies dient heutzutage auch dazu, Verwechslungen mit dem Englischen zu vermeiden.

Beispiel: 27. März 1996

Eingabe: 27.~März 1996

Einige Verlage setzen zur besseren Lesbarkeit bei der Zahlform nach den Punkten einen kleinen Zwischenraum, manchmal sogar vor der Jahreszahl einen Wortzwischenraum, vor allem dann, wenn viele Zahlen im Text vorkommen.

Beispiel: 27. 3. 1996, 4. 1. 1997

Eingabe: 27.\, 3.\, 1996,
4.\, 1.~1997

Die Angabe von 0 für fehlende Tages- und Monatsstellen ist nur in juristischen Abhandlungen und in Tabellen üblich.

Beispiel: 27.03.1996

Die Datumsform ist stark landespezifisch. Bei gemischtsprachigen Texten kann daher die Angabe nach ISO-Norm sinnvoll sein.

Beispiel: 1996–03–27

- Zur Zusammenfassung aufeinanderfolgender oder aus der Geschichte geläufiger Jahreszahlen verwendet man den Schrägstrich.

Beispiel: 1933/45

- Bei Zeitangaben werden die Minuten von den Stunden durch einen Punkt oder durch Hochstellung abgehoben.

Beispiel: 8.30 Uhr, 10³⁰ Uhr

Eingabe: 10\ (^{30}\)~Uhr oder
10\${}^{30}\$~Uhr

Zeitangaben sind landespezifisch. Bei gemischtsprachigen Texten kann daher eine Angabe nach ISO-Norm sinnvoll sein.

Beispiel: 8:35, 19:47:11

15.8 Nummern

- Telefon-, Fax- und Postfachnummern werden, von der letzten Ziffer ausgehend, in Zweiergruppen gegliedert. Die Ortsnetzkennzahl wird für sich ebenso gegliedert und in runde Klammern gesetzt oder durch einen Schrägstrich abgetrennt.

Beispiel: (0 62 21) 2 97 66,
0 62 21/16 79 06

Eingabe: (0\, 62\, 21)~2\, 97\, 66,
0\, 62\, 21/16\, 79\, 06

Ausnahme: In der Schweiz werden bei siebenstelligen Telefonnummern die ersten drei Ziffern zusammengefasst, die Ortsnetzkennzahl wird nicht gegliedert.

Beispiel: (064) 922 71 31,
064/922 71 31

Eingabe: (064)~922\, 71\, 31
064/922\, 71\, 31

Zur Verdeutlichung kann bei Telefonanlagen mit mehreren Apparaten der Bindestrich verwendet werden.

Beispiel: (0 62 21) 54-72 67

Eingabe: (0\, 62\, 21)\, 54-72\, 67

Die Landesvorwahl kann mit einem Pluszeichen (+) vorangestellt werden, wobei dann normalerweise die 0 für die Ortsnetzkennzahl entfällt.

Beispiel: +49 (62 21) 2 97 66, +49/62 21/16 79 06

- Für Postleitzahlen lautete die Regelung bei den vierstelligen Ziffern, dass sie nicht gegliedert werden. Für die fünfstelligen Ziffern ist diese Richtlinie nicht geändert worden.

- Bis zu vierstellige DIN-Nummern werden nicht getrennt. Für fünf- und mehrstellige DIN-Nummern wird eine Gliederung wie bei Zahlen vorgenommen. Zwischen der Abkürzung DIN und der Nummer darf kein Zeilenumbruch erfolgen.

Beispiel: DIN 6654, DIN 41 570

Eingabe: DIN~41\, 570

- Sonstige Nummern, wie beispielsweise Kunden- oder Kontonummern können anders gruppiert werden.

Beispiel: 2134 00-757, 23 100 07, 670 601 29

15.9 Striche

Es gibt zwei verschiedene Typen von Strichen: den Bindestrich, auch Trennstrich oder Divis genannt, und den Gedankenstrich oder Halbgeviertstrich. (Das Minuszeichen als dritter Strichtyp wird im Abschnitt „Mathematischer Satz“, S. 569, behandelt.)

- Der Bindestrich wird für Bindungen und Trennungen verwendet. Zusammengesetzte Wörter werden gewöhnlich ohne Bindestrich geschrieben. Ausnahmefälle sind z. B. unübersichtliche Zusammensetzungen aus mehr als drei Gliedern und Zusammensetzungen mit Abkürzungen oder fremdsprachigen Wörtern.

Beispiel: Pkw-Ersatz, Après-Ski

Bindestriche werden auch eingesetzt bei Ergänzungen, zur Verdeutlichung und bei Aneinanderreihungen.

Beispiel: Müller-Schulze, bergauf und -ab, Ein-/Ausgang,
Formel-1-Rennwagen, Primär-(Haupt-)Stromkreis

Eingabe von Binde- und Trennstrichen (siehe auch den Abschnitt über Worttrennungen, S. 572):

- Trennstriche bei Worttrennungen am Zeilenende werden von $\TeX$ automatisch mit dem richtigen Zeichen eingefügt. Bei der Eingabe von Bindestrichen als Ergänzungszeichen ist darauf zu achten, dass nach einem Bindestrich (bei Standardeinstellungen) immer eine mögliche Trennstelle gefunden wird. Die Eingabe von "~ erzeugt einen Bindestrich, nach dem nicht getrennt werden darf.

Beispiel: bergauf und -ab

Eingabe: bergauf und "~ab

- Die Eingabe von "=" erzeugt einen Bindestrich, nach dem immer eine Trennstelle gefunden wird, die aber sehr nahe am Bindestrich liegen kann.

Beispiel: Unfall-Versicherung Eingabe: Unfall"=Versicherung

- Oft ist es sinnvoll, eine andere Trennstelle im Wort anzugeben, die aber ohne Einfügen eines Trennstrichs getrennt werden soll. Dazu dient die Eingabe von "~".

Beispiel: Primär-(Haupt-) Eingabe: Prim"ar-(Haupt"~) "%
Stromkreis, Ein-/ Stromkreis, Ein"~/ "Ausgang
Ausgang

- Der Gedankenstrich dient

- dem Einschub von Sätzen mit Wechsel des Themas oder des Sprechers,
- dem Einschub von Satzstücken oder Sätzen, die das Gesagte erläutern oder ergänzen,
- innerhalb eines Satzes zur Kennzeichnung einer längeren Pause.

Beispiel: „Mein Sohn, was birgst du Eingabe: ... -- ...
so bang dein Gesicht“ – „Siehst,
Vater, du den Erbkönig nicht?“

Wenn der Gedankenstrich zum Einschub von Text dient, wird er in der Regel zu Beginn und am Ende des Einschubs gesetzt. Ist das Ende des Einschubs gleichzeitig das Satzende, so entfällt der Gedankenstrich vor dem Satzpunkt. Ein Zeilenumbruch nach dem „öffnenden“ und vor dem „schließenden“ Gedankenstrich sollte, besonders bei kurzen Einschüben, vermieden werden. Dazu wird in der Eingabe ~ statt eines Leerzeichens verwendet.

Beispiel: Der Autor – er ist Eingabe: Der Autor --~er ist
Spezialist – betont ... Spezialist~-- betont ...

In amerikanischen Texten ist der Gedankenstrich oder auch *em-dash* länger und wird ohne Zwischenraum gesetzt.

Beispiel: ... they were modern, Eingabe: ... modish---use ...
stylish, modish—use whatever
term you will.

- Der Gedankenstrich findet neben seiner eigentlichen Bestimmung auch als Zeichen für „gegen“ und „von-bis“ Verwendung.

- Der Gegen-Strich ist wie auch der Gedankenstrich von Wortzwischenraum umgeben.

Beispiel: Schalke 04 – Eintracht Eingabe: Schalke~04~--
Frankfurt Eintracht Frankfurt

- Der Von-Bis-Strich wird ohne Zwischenraum gesetzt.

Beispiel: 8–11 Uhr, Eingabe: 8--11~Uhr
Hauptstr. 47–51, Paris–Dakar

Anmerkung: In T1-kodierten Schriften gibt es noch zwei weitere Striche. Sie können unter Verwendung des Pakets `textcomp` mit `\textttwelveudash` (Zweidrittel-Geviertstrich) und mit `\textttthreequartersemdash` (Dreiviertel-Geviertstrich) gesetzt werden.

Striche

Bindestrich, Divis, <i>hyphen</i> (6 Einheiten lang)	- -
Gedankenstrich, Halbgeviertstrich, Von-Bis-Strich, <i>en-dash</i> (9 Einheiten lang)	- -
Amerikanischer Gedankenstrich, Geviertstrich, <i>em-dash</i> (18 Einheiten lang)	— --
Zweidrittel-Geviertstrich, Zwölfeinheitenstrich (12 Einheiten lang)	— \textttwelveudash{}
Dreiviertel-Geviertstrich (13,5 Einheiten lang)	— \textttthreequartersemdash{}
Minuszeichen	- \(-\), \$-\$

15.10 Klammern, geschachtelte Klammern

Im Wesentlichen werden Klammern in zwei Fällen eingesetzt: Zum einen bei erklärenden Zusätzen und Zusätzen innerhalb von Wortverbindungen

Beispiel: Oggersheim (Pfalz), Gemeinde(amts)vorsteher

und zum anderen bei eingeschobenen Sätzen, die ohne Nachdruck gesprochen werden (an Stelle von Kommas oder Gedankenstrichen). In wissenschaftlichen Texten werden Klammern (eckige oder runde) außerdem noch zur Angabe von Quellen und Literaturverweisen verwendet.

15.10.1 Schreibweise von Klammern

- Vor der öffnenden und nach der schließenden Klammer wird ein Leerzeichen gesetzt. Ausnahmen hiervon sind schließende Klammern vor Satzzeichen. Zwischen den Klammern und dem von ihnen eingeschlossenen Text steht kein Zwischenraum.
- Schachtelung von Klammern sollten vermieden werden. Wenn notwendig, kann man zwischen runden und eckigen Klammern abwechseln.

Beispiel: Mit dem Wort Bankrott (vom italienischen „banca rotta“ [zusammengebrochene Bank]) bezeichnet man die Zahlungsunfähigkeit.

Im Text sind eckige Klammern den runden Klammern übergeordnet. Dies gilt jedoch nur unter Berücksichtigung der Metabedeutung von Klammern. Werden, wie im obigen Beispiel, Anmerkungen immer in runde Klammern gesetzt, so sind eckige Klammern den runden Klammern untergeordnet.

Bei Literaturverweisen in eckigen Klammern sind runde Klammern übergeordnet.

- Werden Klammern zur Angabe von Quellen bei Zitaten verwendet, so wird, wenn die Angabe bei Textteilen erfolgt, die selbst bereits in Klammern eingefasst sind, diese Quellenangabe durch Kommas anstelle weiterer Klammern vom umgebenden Text abgesetzt.

Beispiel: Dies ist ein Beispiel (Niemand, 1911). Außergewöhnlich . . . (siehe auch die Verteilung in Tab. 1 von Müller & Meier, 1993, als weiteren Beleg).

15.11 Wortzwischenräume, Abstände (Spationierung)

- Wortzwischenraum wird durch Eingabe eines Leerzeichens erzeugt. (Die Eingabe von mehr als einem Leerzeichen bleibt bei $\text{\LaTeX}/\text{\TeX}$ wirkungslos.) Die Breite des Wortabstands und seine Variabilität, um die Zeilen optimal aufzufüllen, sind mit Parametern steuerbar.
- Im Amerikanischen wird nach einem Satzendezeichen ein vergrößerter Zwischenraum gesetzt. Stellt ein Punkt kein Satzendezeichen dar und ist das Zeichen davor ein Kleinbuchstabe, muss durch Eingabe von `\_` der vergrößerte Abstand verhindert werden.

Beispiel: e.g. $\text{\TeX}$

Eingabe: e.g. `\_\text{\TeX}`{ }

Andererseits wird bei einem Großbuchstaben, dem ein Punkt folgt, automatisch ein Wortzwischenraum und kein vergrößerter Zwischenraum gesetzt. Hier muss durch Eingabe von `\@` der vergrößerte Abstand erzwungen werden.

Beispiel: The sign of Zorro is a Z.

Eingabe: ... is a Z`\@`.

Im Deutschen ist der Zwischenraum nach einem Satzendezeichen ebenso groß wie zwischen Worten. Das Paket `german` ruft automatisch den Befehl zum Abschalten vergrößerter Zwischenräume auf: `\frenchspacing`.

- Bei jedem Leerzeichen kann ein Zeilenumbruch erfolgen. Die Tilde (~) wirkt als *tie*, also als Band/Knoten zwischen zwei Zeichen, mit der Folge, dass die beiden Zeichen immer in einer Zeile stehen.

Beispiel: Bd. 1

Eingabe: Bd.`\~`1

- Ein „kleiner Wortzwischenraum“, kurz „kleiner Zwischenraum“ (Fachausdruck: *Spatium*) – englisch *thin space* –, wird beispielsweise bei mehrgliedrigen Abkürzungen gesetzt. Durch Eingabe von `\,` bei $\text{\LaTeX}$ wird er erzeugt.

Beispiel: z. B.

Eingabe: z.`\,`B.

Bei Plain- $\text{\TeX}$ funktioniert dieser Befehl nur im *math mode*, für den *text mode* lautet er `\thinspace`.

- Wird kursive und geradestehende Schrift im Text verwendet, so muss beim letzten geneigten Buchstaben eine Abstandskorrektur erfolgen, damit er nicht in den nachfolgenden geraden Text bzw. Abstand hineinragt. Dies wird von $\text{\LaTeX}$ in den meisten Fällen richtig gemacht, wenn man den Befehl `\emph` verwendet.

Beispiel: Ein *dreiviertel*volles Glas Eingabe: ... ist nicht
ist nicht *dreiviertel*leer. `\emph{dreiviertel}leer.`

Beispiel: Zur Abschreckung ohne Abstandskorrektur dient die *Kabellänge*.

Bei $\text{\LaTeX}$ 2.09 muss die Korrektur bei den Befehlen `\it`, `\em` und `\sl` durch Eingabe von `\` gesetzt werden.

Beispiel: *dreiviertel*leer Eingabe: `{\it dreiviertel\}`
leer

- Der Einzug am Absatzanfang sollte die Breite eines Gevierts haben, denn es wird ein Einzug als günstig empfunden, der seitlich wie in der Höhe gleich ist, also ein optisches Quadrat bildet. Ein Geviert oder Druckerviertel ist eine Längeneinheit, die von der verwendeten Schriftart und -größe bestimmt wird. Es hat ungefähr die Breite des Buchstabens „M“ (Einheit 1 em). Der Standardeinzug in $\text{\LaTeX}$ ist etwas breiter: 20 pt. Der zugehörige Längenparameter in $\text{\TeX}$ bzw. $\text{\LaTeX}$ heißt `\parindent`.
- Bei $\text{\TeX}$ ist der Zeilenabstand (Durchschuss) fix. Um eine Seite vertikal optimal aufzuteilen, steckt Variabilität im Abstand von Absätzen sowie vor und nach Überschriften und Formeln. Auch hier erfolgt die Steuerung über Parameter (siehe beispielsweise Goossens et al. (1995) unter dem Stichwort „Stilparameter“).

Zwischenräume

Wortzwischenraum (Leerzeichen)	<code>\</code>
Geschütztes Leerzeichen	<code>~</code>
Kleiner Zwischenraum, Spatium, $\frac{3}{18}$ quad	<code>\,</code>
Kleiner negativer Zwischenraum, $-\frac{3}{18}$ quad	<code>\(!\)</code>
Großer Zwischenraum	<code>\quad</code>
Noch größerer Zwischenraum	<code>\qquad</code>

15.12 Mathematischer Satz

Beim mathematischen Satz übernimmt $\text{\TeX}$ den Satz von fast allen Feinheiten, insbesondere allen Abständen, die es zu beachten gilt, wenn man Zahlen und Rechenzeichen den typografischen Regeln gemäß setzt.

Eingabe: `\ (... \)` oder `\$...\$` und für abgesetzte Formeln `\[...\]` oder `\$\$...\$\$`

Im Folgenden wird nur auf einige oft gemachte Fehler eingegangen, die häufig in Texten aus dem geisteswissenschaftlichen Bereich mit nur wenigen Formeln zu beobachten sind.

- Ein Satz sollte nicht mit einer Formel oder Zahl beginnen.
- Es sollten nie zwei Formeln direkt oder nur durch ein Satzzeichen getrennt nebeneinander stehen.

Beispiel: Statt $a + b, c + d$ besser $a + b$ und $c + d$

- Das Minuszeichen muss wie ein Pluszeichen ohne senkrechten Strich aussehen, ist also nicht zu verwechseln mit dem Binde- oder Gedankenstrich.

Beispiel: -293

Eingabe: `\(-293\)` oder `\(-\)293`

- Als Multiplikationszeichen zwischen Zahlen wird oft das Zeichen $\times$ verwendet, welches 1631 von Oughtred eingeführt wurde. Es unterscheidet sich ganz erheblich von dem oft statt dessen verwendeten Buchstaben x .

Beispiel: 2×2 -Design, $9\text{ cm} \times 12\text{ cm}$ Eingabe: `\(2 \times 2\)`-Design

Im englischen Sprachraum wird dieses Zeichen außerdem noch für Zehnerpotenzen verwendet, im Deutschen ist dies nicht erwünscht.

Beispiel: 1.245×10^{-3}

Zwischen Variablen wird grundsätzlich kein Multiplikationszeichen gesetzt. Gegebenenfalls kann man zur Verdeutlichung den etwas erhöht auf der Zeile stehenden Malpunkt verwenden.

Beispiel: $a \cdot b$

Eingabe: `\(a \cdot b\)`

Der Stern als Multiplikationszeichen kommt ausschließlich in der elektronischen Datenverarbeitung vor.

Beispiel: $a * b$

Eingabe: `\(a \ast b\)`

- Der Abstand bei einem Gleichheitszeichen wird von $\text{\TeX}$ nur dann richtig berechnet, wenn vor und nach dem Gleichheitszeichen (innerhalb der mathematischen Umgebung) ein Wert steht. Bei Abwesenheit eines Werts kann man durch Eingabe von `\{ }` den richtigen Abstand erwirken.

Beispiel: $1 + 1$ ist nicht immer $= 2$ Eingabe: `\(1+1\)` ... immer `\(\{ \}=2\)`

- Innerhalb des mathematischen Satzes werden alle Buchstaben als Variablen betrachtet und deshalb kursiv, ohne Ligaturen und mit einem bestimmten Abstand zueinander gesetzt. Will man nun Text innerhalb einer Formel schreiben, so muss eine spezielle Schriftart gewählt werden.

Beispiel: Effektivzins $= 6\%$

Eingabe: `\(\mathrm{Eff}...\)`
`= 6\,\%\)` (mit dem Paket `amstext`:
`\(\text{Eff}... = 6\,\%\)`)

- Werden in einer Formel bei einer Variablen Hoch- und Tiefstellungen (*super-* und *subscript*) zusammen und bei einer anderen nur eines von beiden verwendet, so sind die Indizes nicht in einer Linie. Eine Korrektur erfolgt durch Eingabe von `\{ }` für den fehlenden Wert.

Beispiel: $a_l^2 b_m c_n^2$ vs. $a_l^2 b_m c_n^2$

Eingabe: `\[a_1^2 b_m^{\{ } c_n^2\]`

- Brüche lassen sich auf verschiedene, äquivalente Weisen schreiben:

$$\frac{a}{b} \quad a/b \quad ab^{-1} \quad a \div b.$$

Das Zeichen $\div$ ist im englischen Sprachraum und in der Datenverarbeitung verbreitet. Es wird oft als Pellsches Divisionszeichen bezeichnet.

Der Bruch $\frac{\pi}{2}$ darf nie mit Schrägstrich gesetzt werden.

Im Fließtext sieht ein Bruch mit Zahlen in der Form $\frac{1}{2}$ schöner aus, was mit dem nachfolgenden kleinen Makro aus (Knuth, 1986e, S. 311) erzeugt wurde:

```
\newcommand{\nfrac}[2]{\leavevmode\kern.1em%
  \raise.5ex\hbox{\scriptsize #1}%
  \kern-.1em/\kern-.15em%
  \lower.25ex\hbox{\scriptsize #2}}
```

Beispiel: $\frac{3}{4}$

Eingabe: `\(\nfrac{3}{4}\)`

- Die Reihenfolge von Klammerebenen im mathematischen Satz ist $\{[()]\}$.
- Beim mathematischen Satz fügt $\text{\TeX}$ hinter einem Dezimalkomma einen kleinen Zwischenraum ein. Der Grund dafür ist, dass das amerikanische Dezimaltrennzeichen der Punkt ist und Kommas daher nur als Trennzeichen bei Aufzählungen verwendet werden, wie beispielsweise in $f(x, y)$. Zur Korrektur umschließt man das Dezimalkomma mit geschweiften Klammern.

Beispiel: $e \approx 2,718$ statt $e \approx 2,718$

Eingabe: `\(\mathrm{e} \ \approx 2\{, \}718\)`

Das englische Dezimaltrennzeichen ist übrigens ein hochgesetzter Punkt, meist wird jedoch der Punkt verwendet.

- Die Schreibweise von Vorsilben bei mathematischen Bezeichnungen bereitet immer wieder Kopfzerbrechen. Vorsilben wie „anti“, „bi“ und „semi“ werden nur dann mit Bindestrich geschrieben, wenn wieder ein „i“ folgt, für „pseudo“ gilt dies analog, wenn ein „o“ folgt.

Beispiel: Pseudodivision; aber: Pseudo-Operator

Für die Vorsilbe „nicht“ scheint es keine generelle Regel zu geben; man sollte gegebenenfalls in einem mathematisch-naturwissenschaftlichen Lexikon nachschlagen.

Beispiel: nichtrandomisierter Test, nicht singuläre Matrix

Mathematischer Satz

Formel im Text		<code>\(\)</code> oder <code>\\$ \\$</code>
Formel abgesetzt		<code>\[\]</code> oder <code>\$\$ \$\$</code>
Minuszeichen	-293	<code>\(-293\)</code>
Kreuzzeichen	2×2	<code>\(2 \times 2\)</code>
Gleichheitszeichen	$\text{ist} = 2$	<code>ist\(\{ \}=2\)</code>
Brüche	$\frac{a}{b}$	<code>\(\frac{a}{b}\)</code>
	$a/b, ab^{-1}$	<code>\(a/b\), \((ab^{-1})\)</code>
	$a \div b$	<code>\(a \div b\)</code>
	$\frac{1}{2}$	<code>\(\nfrac{1}{2}\)</code>
Dezimalkomma	$2,718$	<code>\(2\{, \}718\)</code>

15.13 Worttrennungen

Trennungen müssen immer überprüft werden! Es gibt richtige, aber den Lesefluß störende Trennungen, und falsche Trennungen. Besonders gefährdet sind zusammengesetzte Wörter, wie sie im Deutschen häufig verwendet werden.

Beispiel: Ur|in|stinkt, Er|stin|stanz, Ver|bein|sparung, Rechts|schut|z|in|ter|es|se
(die von $\TeX$ gefundenen Trennstellen sind durch | angegeben)

Anmerkung: Durch den Einsatz von T1-kodierten Schriften statt der OT1-kodierten Schriften verschwinden etliche falsche Trennungen, außerdem werden erheblich mehr Trennstellen, insbesondere bei Wörtern mit Umlauten, gefunden. Zur Verwendung von T1-kodierten Schriften muss in der Präambel des Dokuments der folgende Befehl stehen:

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

- Trennungen ermöglichen:

- "- zusätzliche zu den von $\TeX$ erkannten Trennstellen,
- \- nur diese Trennstelle ist im Wort erlaubt, verhindert damit weitere, evtl. falsche Trennungen,
- "" zusätzliche zu den von $\TeX$ erkannten Trennstellen, bei der kein Trennstrich im Trennungsfall eingefügt wird,
- "= Bindestrich zusätzlich zu den von $\TeX$ erkannten Trennstellen, bei dem jedoch nahe am Bindestrich eine weitere mögliche Trennstelle gefunden werden kann.

Anmerkung: Die generelle Verwendung von "=" ist nicht empfehlenswert, $\TeX$ findet dadurch möglicherweise unschöne Trennstellen.

Beispiel: Worttrenn-Feh|ler

Eingabe: Worttrenn"=Fehler

`\hyphenation{...}` Ausnahmelexikon. Die Wörter werden mit allen möglichen Trennstellen angegeben, so dass sie im gesamten Text getrennt werden können.

Eingabe: `\hyphenation{Tu-to-ri-um}`

Anmerkung: Nur mit T1-kodierten Schriften können Wörter mit Sonderzeichen wie Umlaute in das Ausnahmelexikon aufgenommen werden.

- Trennungen verhindern:

- "~ Bindestrich ohne Trennstelle,

`\mbox{...}` Verhinderung von Trennungen innerhalb eines Wortes,

Beispiel: BVerfGE

Eingabe: `\mbox{BVerfGE}`

`\hyphenation{...}` Ausnahmelexikon. Die Wörter werden ohne Trennstellen angegeben, so dass sie im gesamten Text nicht getrennt werden können.

Eingabe: `\hyphenation{Hallo}`

Anmerkung: Nur mit T1-kodierten Schriften können Wörter mit Sonderzeichen wie Umlaute in das Ausnahmelexikon aufgenommen werden.

- Sonstige Trennungen (nur bei alter Rechtschreibung):

Ist bei einer Zusammensetzung an der Wortfuge einer von drei gleichen Konsonanten weggefallen, so tritt der Konsonant bei der Silbentrennung wieder ein.

Beispiel: Schiff-fahrt, Roll-laden, Brenn-nessel, Ballett-theater; Eingabe: "ff für „ff“, welches bei Trennung zu ff-f wird, ebenso "ll, "mm, "nn, "pp, "rr und "tt

Die Konsonantenverbindung „ck“ wird bei Silbentrennung in k-k aufgelöst. Bei Namen sollte die Trennung von ck möglichst vermieden werden.

Beispiel: Druk-kerei

Eingabe: `Dru"ckerei`

("ck für ein „ck“, welches bei Trennung zu k-k wird.)

- Verwendungsbeispiele:

Die möglichen Silbentrennstellen sind in den Beispielen durch das Zeichen | gekennzeichnet (aus Raichle (1998))

Auf" lage	ergibt:	Auflage (statt Auflage) mit den Trennstellen: Auf la ge
"ubertragen	ergibt:	übert ra gen (falsche Trennstelle!)
"uber\~tragen	ergibt:	über tragen
"uber"-tragen	ergibt:	über tra gen
bergauf und -ab	ergibt:	berg auf und - ab (falsche Trennstelle)
bergauf und "~ab	ergibt:	berg auf und -ab
I-Punkt	ergibt:	I- Punkt (schlechte Trennstelle)
I"~Punkt	ergibt:	I-Punkt
Arbeiter"=Unfall	ergibt:	Ar bei ter- Un fall ver si che rungs-
versicherungsgesetz		ge setz

- Tipps zur Vorgehensweise beim Trennen (aus Raichle (1998)):
 - Bei zusammengesetzten Wörtern, die falsche Trennstellen aufweisen, sollte man zuerst die Trennhilfe "– in die Wortfuge der Zusammensetzung einfügen und nochmals testen, ob danach richtig getrennt wird, bevor man weitere Trennhilfen einfügt.
 - Bei Verwendung von Bindestrichen zur Ergänzung („bergauf und -ab“, „ein-, zwei- oder dreimal“), vor Schrägstrichen („Ein-/Ausgang“) und bei Klammern („Primär-(Haupt-)Strom“) sollte man mit den Befehlen `\mbox`, `" "` und `"~` vor den Satzzeichen die Trennung unterbinden und evtl. nach den Satzzeichen wieder erlauben. (Letzteres ist bei Schrägstrichen unbedingt notwendig.)
 - Den Befehl `"=` sollte man nur für automatisch erstellte Texte verwenden, da er in den mit dem Bindestrich verbundenen Wortteilen auch nahe am Bindestrich liegende Trennstellen erlaubt, die man vermeiden sollte. Zusammengesetzte Wörter sollten grundsätzlich ohne Bindestrich geschrieben werden. Wenn der Bindestrich dennoch zur Vermeidung von Missverständnissen oder zur Verdeutlichung notwendig ist, sollte man mit `\–` und `"–` wenige wohlüberlegte Trennhilfen einfügen.

Trennungen

Zusätzl. Trennung	<code>"–</code>
Einzigste Trennung	<code>\–</code>
Zusätzl. Trennung ohne Divis	<code>" "</code>
Divis mit Trennung nahe am Divis	<code>"=</code>
Geschützter Divis	<code>"~</code>
Lexikon	<code>\hyphenation{ }</code>
Ohne Trennung	<code>\mbox{ }</code>
Sonstige Trennungen	<code>"ck, "ff, "ll, "mm,</code> <code>"nn, "pp, "rr, "tt</code>

15.14 Zeilenumbruch – *overfull boxes*

Maßnahmen zur Korrektur des Zeilenumbruchs, außer dem Einfügen von zusätzlichen Trennstellen mit `"–` oder `" "`, sollten erst ergriffen werden, wenn der Text inhaltlich steht. Die wichtigsten Parameter, die die

Wortzwischenräume in bündig formatierten Absätzen beeinflussen, sind `\tolerance` und `\emergencystretch`; Genaueres dazu findet sich in (Goossens et al., 1995, "Ausrichten von Absätzen", S. 51 f.). Die Befehle `\fussy` und `\sloppy` beeinflussen diese beiden Parameter, wobei letzterer relativ ungünstige Zeilenumbrüche zulässt. Besser ist es, die beiden Parameter selbst zu setzen und sich dem Optimalwert für einen einzelnen Text durch Probieren anzunähern.

Neben den genannten Parametern gibt es noch Parameter für jeden einzelnen Zeichensatz (vgl. (Goossens et al., 1995, "Einbindung neuer Zeichensätze", S. 206)).

Als allerletztes Mittel bleibt noch die Änderung der Satzstellung oder Umformulierungen des Textes, um eine bessere Zeilenfüllung zu erreichen.

15.15 Spezielle Literatur

- [1] Friedrich L. Bauer: *Punkt und Komma*. In: Informatik Spektrum, 19(2): 93–95, April 1996.
- [2] Karl Billmann und H. Nacke: *Manuskript-Richtlinien*. Werner Verlag, Düsseldorf, 2. Aufl., 1986.
- [3] *Duden „Deutsches Universalwörterbuch A–Z“*. Bibliographisches Institut, Mannheim, 2. Aufl., 1989.
- [4] Klaus Poenicke: *Duden „Die schriftliche Arbeit“: Materialsammlung und Manuskriptgestaltung für Facharbeiten, Seminararbeiten und Abschlussarbeiten an Schule und Universität*. Bibliographisches Institut, Mannheim, 2., verb. Aufl., 1989.
- [5] Friedrich W. Weitershaus: *Satz- und Korrekturanweisungen: Richtlinien für die Textfassung*. Duden-Taschenbücher Band 5, Bibliographisches Institut, Mannheim, 6. Aufl., 1994 (vergriffen, Neuaufgabe unbestimmt).
- [6] Josef Werlin: *Wörterbuch der Abkürzungen*. Duden-Taschenbücher Band 11, Bibliographisches Institut, Mannheim, 3., neubearb. und erw. Aufl., 1987.
- [7] *Duden „Rechtschreibung der deutschen Sprache und der Fremdwörter“*. Hrsg. von der Duden-Redaktion auf der Grundlage der amtlichen Rechtschreibregeln. Bibliographisches Institut, Mannheim, 20. Aufl., 1991.

- [8] Oliver Heidelberg: *V.E.R.A. – Verzeichnis EDV-Relevanter Akronyme*. <http://userpage.fu-berlin.de/öheiabbd/FTP>
- [9] Harald Kipp: *Lexikon der europäischen Abkürzungen*. Bechtermünz, Eltville, 1990.
- [10] Hildebert Kirchner: *Abkürzungsverzeichnis der Rechtssprache*. de Gruyter, Berlin, 4., ern. und erw. Aufl., 1993.
- [11] Jan Tschichold: *Meisterbuch der Schrift*. Otto Maier Verlag, Ravensburg, 2. neubearbeitete Auflage, 1965, S. 42 ff.

A

Von L^AT_EX 2.09 nach L^AT_EX 2_ε

Es kann immer mal wieder vorkommen, dass ein Text auftaucht, der noch mit L^AT_EX 2.09 gesetzt ist. Man erkennt einen solchen Text daran, dass der erste Befehl `\documentstyle` statt `\documentclass` lautet.

Wenn man diesen Text mit der Version L^AT_EX 2_ε bearbeitet, dann ist das kein Problem. Obwohl die beiden Versionen nicht vollständig kompatibel sind, wird jeder Text dieser älteren Version mit L^AT_EX 2_ε im *compatibility mode* bearbeitet. Möchte man diesen Text jedoch überarbeiten, dann ist es sinnvoll, ihn nach L^AT_EX 2_ε zu übertragen. Den Hauptunterschied zwischen beiden Versionen findet man in der Präambel, d. h. vor dem `\begin{document}`. Wie L^AT_EX 2_ε kennt L^AT_EX 2.09 Layouts wie `article`, `report`, `book` usw. und Optionen. Bei den Optionen unterscheidet L^AT_EX 2.09 zwei Arten:

- interne, die aktiviert werden,
- externe, die als Makropaket geladen werden.

Zur Auswahl gibt es den Befehl

```
\documentstyle[<Option(en)>]{<Layout>}
```

Beispielsweise in L^AT_EX 2.09:

```
\documentstyle[twoside,german,lfolsm]{article}
```

Hier wird die Stil-Datei `article` verwendet. Die Option `twoside` ist intern. Für `german` und `lfolsm` werden die Makropakete `german.sty` und `lfolsm.sty` geladen.

L^AT_EX 2_ε vermeidet die Mischung von internen und externen Optionen. Aus `\documentstyle` wird `\documentclass`. Die internen Optionen

werden bei `\documentclass` angegeben; für die externen Optionen, d. h. das Laden der Makropakete, wird der Befehl `\usepackage` verwendet.

Damit kann das Beispiel wie folgt nach L^AT_EX 2_ε übertragen werden:

```
\documentclass[twoside]{article}
\usepackage{german}
\usepackage{lfolsm}
```

Die meisten externen Optionen, d. h. die L^AT_EX 2.09 Makropakete, arbeiten auch unter L^AT_EX 2_ε. Da die Möglichkeiten von L^AT_EX 2.09 erweitert wurden, ist es das Einfachste, nach diesen Änderungen den Text mit L^AT_EX 2_ε zu formatieren und dann eventuell auftretende Fehler (falls überhaupt welche auftauchen) zu beseitigen.

B

Seitenlayout

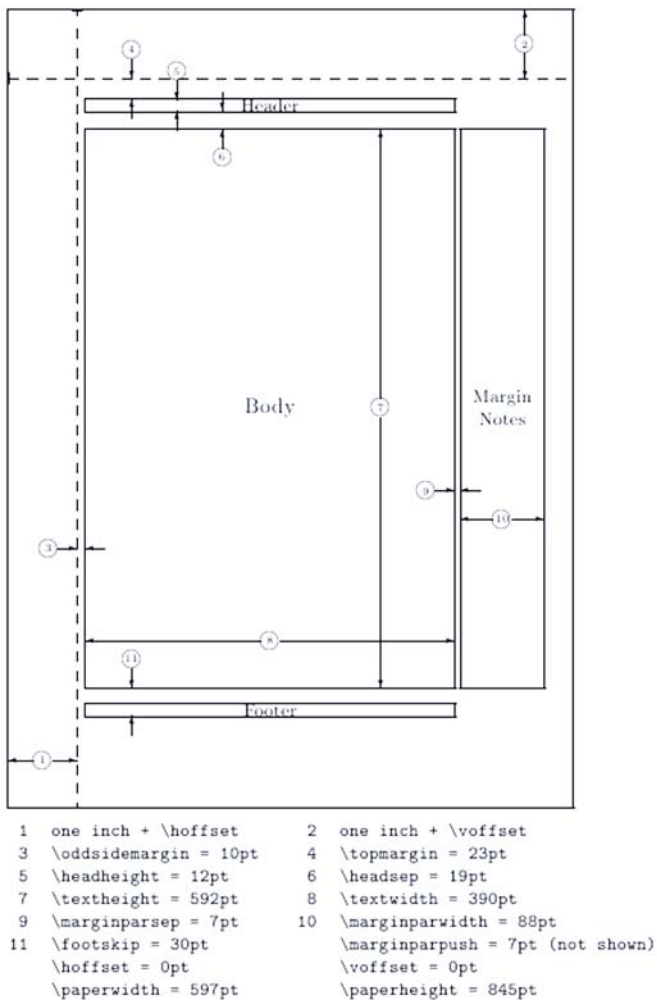


Abbildung B.1: Layout einer rechten/ungeraden Seite

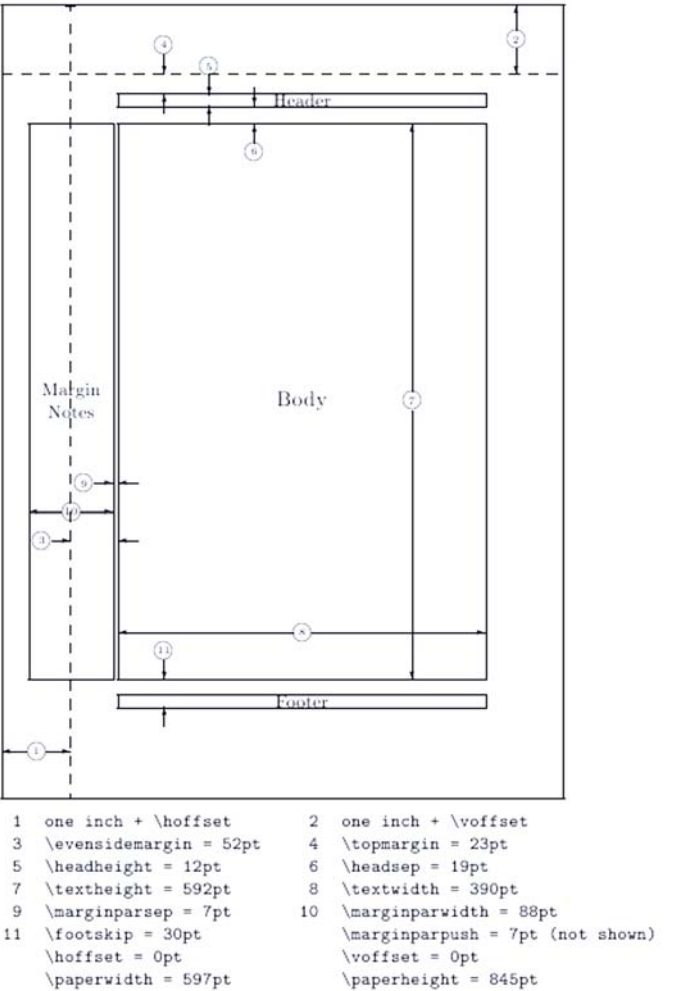


Abbildung B.2: Layout einer linken/geraden Seite

C

Dateinamenserweiterungen

Da in diesem Buch $\text{\LaTeX} 2_{\epsilon}$ das Thema ist, wird nicht zwischen eigentlichen $\text{\TeX}$ - und echten $\text{\LaTeX}$ -Dateien unterschieden, die Reihenfolge entspricht der Wichtigkeit für den Anwender:

- .tex** Standard-Dateinamenserweiterung für alle Eingabedateien;
- .dvi** $\text{\TeX}$ -Ausgabedatei (DVI für Device Independent), sie enthält das formatierte Dokument – zum Anschauen bzw. Drucken ist ein Treiber nötig;
- .log** Protokoll-Datei eines $\text{\LaTeX}$ -Laufs, hier findet man alle Fehlermeldungen;
- .aux** Hilfsdatei, enthält Querverweise etc. (engl. *auxiliary*);
- .toc** Hilfsdatei für Inhaltsverzeichnis (engl. *table of contents*);
- .lof** Hilfsdatei für Abbildungsverzeichnis (engl. *list of figures*);
- .lot** Hilfsdatei für Tabellenverzeichnis (engl. *list of tables*);
- .ltx** Dokument-Quelldatei, alternativ zur Namensendung **.tex**;
- .sty** Eingabedatei mit Definitionen eines $\text{\LaTeX}$ -Pakets (engl. *style*);
- .dtx** Makro-Quelldatei;
- .dtx** Dokumentations-Quelldatei;
- .cls** Eingabedatei mit Definitionen für die Dokumentklasse (engl. *class*);
- .clo** Eingabedatei mit Definitionen für die Klassen-Optionen der Dokumentklasse (engl. *class option*);
- .cfg** $\text{\TeX}$ -Eingabedatei mit Konfigurationsinformation (engl. *configuration*);

- .**ldf** T_EX-Eingabedatei mit der Sprach-Definition für eine Sprache mit ihren Dialekten des babel-Pakets (engl. *language definition*);
- .**fmt** T_EX-Formatdatei, enthält ein Makropaket.

D

sample2e-Beispiel

Die folgende Eingabe ist seit 1994 Teil jeder $\text{\LaTeX}$ -Verteilung. Sie versucht vieles von dem auszutesten, was $\text{\LaTeX}$ bietet, und stellt gleichzeitig ein Referenzbeispiel dar, wo man man Anregungen für seine eigene Eingabe findet.

```
1
2 % This is a sample LaTeX input file.
3 % (Version of 11 April 1994.)
4 %
5 % A '%' character causes TeX to ignore all
6 % remaining text on the line,
7 % and is used for comments like this one.
8
9 \documentclass{article}
10      % Specifies the document class
11 \setlength{\textwidth}{28pc} % fuer Buch
12 \setlength{\textheight}{540pt} % fuer Buch
13
14      % The preamble begins here.
15 \title{An Example Document}
16      % Declares the document's title.
17 \author{Leslie Lamport}
18      % Declares the author's name.
19 \date{January 21, 1994}
20      % Deleting this command produces today's date.
21
22 \newcommand{\ip}[2]{(#1, #2)}
23      % Defines \ip{arg1}{arg2} to mean
24      % (arg1, arg2).
25
26 %\newcommand{\ip}[2]{\langle #1 | #2\rangle}
27      % This is an alternative definition of
```

```

28             % \ip that is commented out.
29
30 \begin{document} % End of preamble and beginning of text.
31
32 \maketitle      % Produces the title.
33
34 This is an example input file. Comparing it with
35 the output it generates can show you how to
36 produce a simple document of your own.
37
38 \section{Ordinary Text}
39             % Produces section heading. Lower-level
40             % sections are begun with similar
41             % \subsection and \subsubsection commands.
42
43 The ends of words and sentences are marked
44 by spaces. It doesn't matter how many
45 spaces you type; one is as good as 100. The
46 end of a line counts as a space.
47
48 One or more blank lines denote the end
49 of a paragraph.
50
51 Since any number of consecutive spaces are treated
52 like a single one, the formatting of the input
53 file makes no difference to
54 \LaTeX,
55 % The \LaTeX command generates the LaTeX logo.
56 but it makes a difference to you. When you use
57 \LaTeX, making your input file as easy to read
58 as possible will be a great help as you write
59 your document and when you change it. This sample
60 file shows how you can add comments to your own input
61 file.
62
63 Because printing is different from typewriting,
64 there are a number of things that you have to do
65 differently when preparing an input file than if
66 you were just typing the document directly.
67 Quotation marks like
68 ``this''
69 have to be handled specially, as do quotes within
70 quotes:
71 ``\,'this' % \, separates the double and single quote.
72 is what I just
73 wrote, not 'that'\,''.
74
75 Dashes come in three sizes: an
76 intra-word

```

```

77 dash, a medium dash for number ranges like
78     1--2,
79 and a punctuation
80     dash---like
81 this.
82
83 A sentence-ending space should be larger than the
84 space between words within a sentence. You
85 sometimes have to type special commands in
86 conjunction with punctuation characters to get
87 this right, as in the following sentence.
88     Gnats, gnus, etc.\ all
89     % '\ ' makes an inter-word space.
90     begin with G\@.
91     % \@ marks end-of-sentence punctuation.
92 You should check the spaces after periods when
93 reading your output to make sure you haven't
94 forgotten any special cases. Generating an
95 ellipsis
96     \ldots\
97     % '\ ' is needed after '\ldots' because TeX
98     % ignores spaces after command names like \ldots
99     % made from \ + letters.
100    %
101    % Note how a '%' character causes TeX to ignore
102    % the end of the input line, so these blank lines
103    % do not start a new paragraph.
104    %
105 with the right spacing around the periods requires
106 a special command.
107
108 \LaTeX\ interprets some common characters as
109 commands, so you must type special commands to
110 generate them. These characters include the
111 following:
112     \$ \% \& \# \{ and \}.
113
114 In printing, text is usually emphasized with an
115     \emph{italic}
116 type style.
117
118 \begin{em}
119     A long segment of text can also be emphasized
120     in this way. Text within such a segment can be
121     given \emph{additional} emphasis.
122 \end{em}
123
124 It is sometimes necessary to prevent \LaTeX\ from
125 breaking a line where it might otherwise do so.

```

```

126 This may be at a space, as between the ``Mr.'' and
127 ``Jones'' in
128     ``Mr.~Jones'',
129     % ~ produces an unbreakable interword space.
130 or within a word---especially when the word is a
131 symbol like
132     \mbox{\emph{itemnum}}
133 that makes little sense when hyphenated across
134 lines.
135
136 Footnotes\footnote{This is an example of a footnote.}
137 pose no problem.
138
139 \LaTeX\ is good at typesetting mathematical formulas
140 like
141     \(\ x-3y + z = 7 \)
142 or
143     \(\ a_{1} > x^{2n} + y^{2n} > x' \)
144 or
145     \(\ \ip{A}{B} = \sum_{i} a_{i} b_{i} \) .
146 The spaces you type in a formula are
147 ignored. Remember that a letter like
148     $x$ % $ ... $ and \(\ ... \) are equivalent
149 is a formula when it denotes a mathematical
150 symbol, and it should be typed as one.
151
152 \section{Displayed Text}
153
154 Text is displayed by indenting it from the left
155 margin. Quotations are commonly displayed. There
156 are short quotations
157 \begin{quote}
158     This is a short a quotation. It consists of a
159     single paragraph of text. See how it is formatted.
160 \end{quote}
161 and longer ones.
162 \begin{quotation}
163     This is a longer quotation. It consists of two
164     paragraphs of text, neither of which are
165     particularly interesting.
166
167     This is the second paragraph of the quotation. It
168     is just as dull as the first paragraph.
169 \end{quotation}
170 Another frequently-displayed structure is a list.
171 The following is an example of an \emph{itemized}
172 list.
173 \begin{itemize}
174     \item This is the first item of an itemized

```

```

175         list. Each item in the list is marked with
176         a ``tick''.
177         You don't have to worry about what kind
178         of tick mark is used.
179
180     \item This is the second item of the list.
181         It contains another list nested inside it.
182         The inner list is an \emph{enumerated} list.
183     \begin{enumerate}
184         \item This is the first item of an enumerated
185             list that is nested within the itemized
186             list.
187
188         \item This is the second item of the inner list.
189             \LaTeX\ allows you to nest lists deeper
190             than you really should.
191     \end{enumerate}
192     This is the rest of the second item of the outer
193     list. It is no more interesting than any other
194     part of the item.
195     \item This is the third item of the list.
196 \end{itemize}
197 You can even display poetry.
198 \begin{verse}
199     There is an environment
200         for verse \\          % The \\ command separates lines
201     Whose features some poets % within a stanza.
202     will curse.
203
204         % One or more blank lines separate stanzas.
205
206     For instead of making\\
207     Them do \emph{all} line breaking, \\
208     It allows them to put too many words
209     on a line when they'd rather be
210     forced to be terse.
211 \end{verse}
212
213 Mathematical formulas may also be displayed. A
214 displayed formula
215 is
216 one-line long; multiline
217 formulas require special formatting instructions.
218 \[ \ip{\Gamma}{\psi}' = x'' + y^{2} + z_{i}^{n} \]
219 Don't start a paragraph with a displayed equation,
220 nor make one a paragraph by itself.
221

```

Die nächsten 3 Seiten zeigen die Ausgabe des $\text{\LaTeX}$ -Tests sample2e.

An Example Document

Leslie Lamport

January 21, 1994

This is an example input file. Comparing it with the output it generates can show you how to produce a simple document of your own.

1 Ordinary Text

The ends of words and sentences are marked by spaces. It doesn't matter how many spaces you type; one is as good as 100. The end of a line counts as a space.

One or more blank lines denote the end of a paragraph.

Since any number of consecutive spaces are treated like a single one, the formatting of the input file makes no difference to $\text{\LaTeX}$, but it makes a difference to you. When you use $\text{\LaTeX}$, making your input file as easy to read as possible will be a great help as you write your document and when you change it. This sample file shows how you can add comments to your own input file.

Because printing is different from typewriting, there are a number of things that you have to do differently when preparing an input file than if you were just typing the document directly. Quotation marks like “this” have to be handled specially, as do quotes within quotes: “‘this’ is what I just wrote, not ‘that’”.

Dashes come in three sizes: an intra-word dash, a medium dash for number ranges like 1–2, and a punctuation dash—like this.

A sentence-ending space should be larger than the space between words within a sentence. You sometimes have to type special commands in conjunction with punctuation characters to get this right, as in the following sentence. Gnats, gnus, etc. all begin with G. You should check the spaces after periods when reading your output to make sure you haven't forgotten any special cases. Generating an ellipsis ... with the right spacing around the periods requires a special command.

$\text{\LaTeX}$ interprets some common characters as commands, so you must type special commands to generate them. These characters include the following: \$ & % # { and }.

In printing, text is usually emphasized with an *italic* type style.

A long segment of text can also be emphasized in this way. Text within such a segment can be given additional emphasis.

It is sometimes necessary to prevent L^AT_EX from breaking a line where it might otherwise do so. This may be at a space, as between the “Mr.” and “Jones” in “Mr. Jones”, or within a word—especially when the word is a symbol like *itemnum* that makes little sense when hyphenated across lines.

Footnotes¹ pose no problem.

L^AT_EX is good at typesetting mathematical formulas like $x - 3y + z = 7$ or $a_1 > x^{2n} + y^{2n} > x'$ or $(A, B) = \sum_i a_i b_i$. The spaces you type in a formula are ignored. Remember that a letter like x is a formula when it denotes a mathematical symbol, and it should be typed as one.

2 Displayed Text

Text is displayed by indenting it from the left margin. Quotations are commonly displayed. There are short quotations

This is a short a quotation. It consists of a single paragraph of text. See how it is formatted.

and longer ones.

This is a longer quotation. It consists of two paragraphs of text, neither of which are particularly interesting.

This is the second paragraph of the quotation. It is just as dull as the first paragraph.

Another frequently-displayed structure is a list. The following is an example of an *itemized* list.

- This is the first item of an itemized list. Each item in the list is marked with a “tick”. You don’t have to worry about what kind of tick mark is used.
- This is the second item of the list. It contains another list nested inside it. The inner list is an *enumerated* list.
 1. This is the first item of an enumerated list that is nested within the itemized list.
 2. This is the second item of the inner list. L^AT_EX allows you to nest lists deeper than you really should.

This is the rest of the second item of the outer list. It is no more interesting than any other part of the item.

- This is the third item of the list.

¹This is an example of a footnote.

You can even display poetry.

There is an environment for verse
 Whose features some poets will curse.
 For instead of making
 Them do *all* line breaking,
 It allows them to put too many words on a line when they'd rather
 be forced to be terse.

Mathematical formulas may also be displayed. A displayed formula is one-line long; multiline formulas require special formatting instructions.

$$(\Gamma, \psi') = x'' + y^2 + z_i^n$$

Don't start a paragraph with a displayed equation, nor make one a paragraph by itself.

E

Literatur

Neben der im Text zitierten Literatur gibt es eine Reihe von Büchern und Artikeln zu den in diesem Buch beschriebenen Themen, auf die der Leser zurückgreifen kann. Die Reihenfolge folgt subjektiven Gesichtspunkten. Die vollständigen bibliografischen Angaben finden sich im Literaturverzeichnis S. 595.

Referenzliteratur zu T_EX

- [1] *T_EX and METAFONT—New Directions in Typesetting*, Knuth (1979)
Anm.: Eines der ersten Bücher zu T_EX, mit einem Vorwort von G. Bell, damaliger Präsident von DEC; liefert viele Hintergrundinformationen.
- [2] *The T_EXbook*, Knuth (1986e)
- [3] *T_EX—The Program*, Knuth (1986d)
- [4] *The METAFONTbook*, Knuth (1986c)
- [5] *METAFONT—The Program*, Knuth (1986b)
- [6] *Computer Modern Typefaces*, Knuth (1986a)

Englischsprachige Literatur zu T_EX

- [1] *T_EX in Practice*
 - *Basics*, Bechtolsheim (1993a)
 - *Paragraphs, Math and Fonts*, Bechtolsheim (1993b)
 - *Tokens, Macros*, Bechtolsheim (1993c)
 - *Output Routines, Tables*, Bechtolsheim (1993d)
- [2] *T_EX by Topic—A T_EXnian’s Reference*, Eijkhout (1991)

Deutschsprachige Literatur zu T_EX

- [1] *Einführung in T_EX*, Schwarz (1991)
- [2] *T_EX-Tools – Software zum Arbeiten mit T_EX unter Linux*, Braune (1998)

Englischsprachige Literatur zu L^AT_EX

- [1] *L^AT_EX—A Document Preparation System*, Lamport (1994)
- [2] *A Guide to L^AT_EX 2_ε—Document Preparation for Beginners and Advanced Users*, Kopka and Daly (1995)
- [3] *L^AT_EX Line by Line*, Diller (1993)
- [4] *The L^AT_EX Graphics Companion—Illustrating Documents with T_EX and PostScript*, Goossens et al. (1997)
- [5] *The L^AT_EX Web Companion—Integrating T_EX, HTML, and XML*, Goossens and Ratz (1999)
- [6] *An Introduction to L^AT_EX and A_ΛS-L^AT_EX*, Grätzer (1997)
- [7] *L^AT_EX 2_ε – for Authors*, L^AT_EX3 Projekt Team (2001)

Anm.: Beschreibt die Unterschiede zwischen L^AT_EX 2.09 und L^AT_EX 2_ε.

– *for Class and Package Writers*, L^AT_EX3 Projekt Team (2003b)

– *Configuration options*, L^AT_EX3 Projekt Team (2003a)

– *font selection*, L^AT_EX3 Projekt Team (2004)

- [8] *Math Mode*, Voss (2005)
- Anm.: Beschreibt aktuelle Möglichkeiten zum Mathematiksatz inkl. Farbeinsatz.

Deutschsprachige Literatur zu L^AT_EX

- [1] *L^AT_EX 2_ε-Kurzbeschreibung*, Schmidt et al. (1999)
- Anm.: Ersetzt das Buch neben dem Computer.
- [2] *L^AT_EX*
 - *Band 1: Einführung*, Kopka (2000)
 - *Band 2: Ergänzungen*, Kopka (1997)
 - *Band 3: Erweiterungen*, Kopka (1996)
- Anm.: Ab der 2. Auflage L^AT_EX 2_ε; geeignet als Einstieg.

- [3] *LaTeX*
 – *Eine Einführung und ein bisschen mehr . . .*, Jürgens (2000)
 – *Fortgeschrittene Anwendungen*, Jürgens (1995)
 Anm.: Online verfügbare Einstiegsbroschüren.
- [4] *Das LaTeX Handbuch*, Lamport (1995)
- [5] *Schnell ans Ziel mit LaTeX_{2 ϵ}* , Knappen (2004)
 Anm.: Geht auch auf Feinheiten des Textsatzes bzw. mathematischer Formeln ein; Hilfreich für Anfänger und Fortgeschrittene.
- [6] *Der LaTeX Begleiter*, Goossens et al. (1995)
 Anm.: Tipps, Tricks und Beschreibung vieler LaTeX-Makropakete; ungeeignet für Einsteiger.
- [7] *Fragen und Antworten (FAQ) über das Textsatzsystem TeX und DANTE*, Deutschsprachige Anwendervereinigung TeX e. V., Raichle et al. (2003)
 Anm.: Unbedingt lesenswert.
- [8] *KOMA-Script – Eine Sammlung von Klassen und Paketen für LaTeX_{2 ϵ}* , Kohm and Morawski (2003)
 Anm.: Beschreibt für den europäischen Raum angepasste Klassen.
- [9] *LaTeX Praxisbuch*, Niedermair and Niedermair (2004)
- [10] *Layoutänderungen mit LaTeX*, Partl and Kielhorn (1996)
- [11] *LaTeX – Tipps & Tricks: Layoutanpassungen, Programmierung, Grafik, Hilfsprogramme, Zeichensätze*, Klöckl (2002)

LaTeX Online-Hilfe

- [1] *LaTeX_{2 ϵ} Help*, Martinsen et al. (1996)
- [2] *Advanced LaTeX*, Love (1999)
- [3] *LaTeXMath and Graphics*, Love (2004)

Satzkunde

- [1] *Satztechnik und Typografie*
 – *Typografische Grundlagen*, GDP-Autorenkollektiv (1996a)
 – *Satztechnik*, GDP-Autorenkollektiv (1996b)
 – *Avor*, GDP-Autorenkollektiv (1996c)
 – *Formenlehre*, GDP-Autorenkollektiv (1996d)
 – *Schriftkunde*, GDP-Autorenkollektiv (1997)

- [2] *Mut zur Typographie – Ein Kurs für Desktop Publishing*, Gulbins and Kahrmann (2000)
Anm.: Ein Muss für jeden, der sich mit Typografie beschäftigt.
- [3] *Hundertein Sätze zur Buchgestaltung*, Kapr (1975)
- [4] *Buchgestaltung*, Kapr (1963)
- [5] *Manuel de Typographie Française élémentaire*, Perrousseaux (1995)
- [6] *Die Schrift, Geschichte – Gestaltung – Anwendung*, Salberg-Steinhardt (1983)
- [7] *Erfreuliche Drucksachen durch gute Typographie – Eine Fibel für jedermann*, Tschichold (1992)
- [8] *Ausgewählte Aufsätze über Fragen der Gestalt des Buches und der Typographie*, Tschichold (1987)
- [9] *The new Typography*, Tschichold (1995)
- [10] *Ursache & Wirkung – Ein Typographischer Roman*, Spierkermann (1994)
Anm.: Absolut spannend.
- [11] *Erste Hilfe in Typographie – Ratgeber für Gestaltung mit Schrift*, Willberg and Forssman (1999)
Anm.: Knappe und praktische Hilfe; sehr empfehlenswert.
- [12] *Lesetypographie*, Willberg and Forssman (1997)
Anm.: Vergleichende Lösungen für typografische Probleme; ungeeignet für Einsteiger.

Sonstiges

- [1] *Wie man eine wissenschaftliche Abschlußarbeit schreibt*, Eco (1977)
- [2] *Dipl. & Co. – Erstellen wissenschaftlicher Arbeiten*, Beyer and Brixel (1995)
- [3] *Technisches Schreiben*, Rechenberg (2002)
- [4] *Lesezeichen – Kognition, Medium und Materialität im Leseprozess*, Gross (1994)

Literaturverzeichnis

Bechtolsheim, S. v. (1993a). *T_EX in Practice*, Volume I: Basics. New York: Springer-Verlag.

Bechtolsheim, S. v. (1993b). *T_EX in Practice*, Volume II: Paragraphs, Math and Fonts. New York: Springer-Verlag.

Bechtolsheim, S. v. (1993c). *T_EX in Practice*, Volume III: Tokens, Macros. New York: Springer-Verlag.

Bechtolsheim, S. v. (1993d). *T_EX in Practice*, Volume IV: Output Routines, Tables. New York: Springer-Verlag.

Beyer, U. and M. Brixel (1995). *Dipl. & Co. – Erstellen wissenschaftlicher Arbeiten*. Heidelberg: International Thomson Publishing.

Braams, J. (2004). *Babel, a multilingual package for use with L^AT_EX's standard document classes*. Benutzerhandbuch zum Paket babel.

Braune, K. (1998). *T_EX-Tools – Software zum Arbeiten mit T_EX unter Linux*. Heidelberg: dpunkt-Verlag.

DANTE e. V., TUG, Lehmanns Fachbuchhandlung (2005). *T_EX Collection 2005*. DANTE e. V., TUG, Lehmanns Fachbuchhandlung. DVD November 2005.

Diller, A. (1993). *L^AT_EX Line by Line*. Chichester: John Wiley & Sons.

Dudenredaktion (1996). *Duden – Die Rechtschreibung der deutschen Sprache und der Fremdwörter* (22 ed.). Mannheim: Bibliographisches Institut.

Eco, U. (1977). *Wie man eine wissenschaftliche Abschlußarbeit schreibt* (6 ed.). Heidelberg: C. F. Müller. UTB für Wissenschaft Nr. 1512.

Eijkhout, V. (1991). *T_EX by Topic—A T_EXnians Reference*. Wokingham, England: Addison-Wesley Publishing Company. PDF-Version <http://www.cs.utk.edu/~eijkhout/texbytopic.pdf>.

GDP-Autorenkollektiv (1996a). *Satztechnik und Typografie*, Volume 1: Typografische Grundlagen. Bern: Gewerkschaft Druck und Papier, GDP-Verlag.

GDP-Autorenkollektiv (1996b). *Satztechnik und Typografie*, Volume 2: Satztechnik. Bern: Gewerkschaft Druck und Papier, GDP-Verlag.

GDP-Autorenkollektiv (1996c). *Satztechnik und Typografie*, Volume 3: Avor. Bern: Gewerkschaft Druck und Papier, GDP-Verlag.

GDP-Autorenkollektiv (1996d). *Satztechnik und Typografie*, Volume 4: Formenlehre. Bern: Gewerkschaft Druck und Papier, GDP-Verlag.

GDP-Autorenkollektiv (1997). *Satztechnik und Typografie*, Volume 5: Schrifkunde. Bern: Gewerkschaft Druck und Papier, GDP-Verlag.

Goossens, M., F. Mittelbach, and A. Samarin (1995). *Der L^AT_EX Begleiter*. Bonn: Addison-Wesley Publishing Company. Nachdruck 2000.

Goossens, M. and S. Ratz (1999). *The L^AT_EX Web Companion—Integrating T_EX, HTML, and XML*. Reading, Massachusetts: Addison-Wesley-Longman.

Goossens, M., S. Ratz, and F. Mittelbach (1997). *The L^AT_EX Graphics Companion—Illustrating Documents with T_EX and PostScript*. Reading, Massachusetts: Addison-Wesley-Longman.

Gross, S. (1994). *Lesezeichen – Kognition, Medium und Materialität im Leseprozess*. Darmstadt: Wissenschaftliche Buchgesellschaft.

Grätzer, G. (1997). *An Introduction to L^AT_EX and A_MS-L^AT_EX*. Bosten: Birkhäuser.

Gulbins, J. and C. Kahrman (2000). *Mut zur Typographie – Ein Kurs für Desktop Publishing* (2 ed.). Heidelberg: Springer-Verlag.

Jones, D. M. (2004). User's guide to the `amsrefs` package. Technical report, American Mathematical Society, Providence, RI. (Version 2.0). Anm.: Benutzerdokumentation zu den AMS-Erweiterungen für Literaturangaben.

Jürgens, M. (1995). *L^AT_EX – Fortgeschrittene Anwendungen*. FernUniversität Hagen, Universitätsrechenzentrum. <ftp://ftp.fernuni-hagen.de/pub/pdf/urz-broschueren/broschueren/a0279510.pdf>.

Jürgens, M. (2000). *L^AT_EX – Eine Einführung und ein bisschen mehr . . .*. FernUniversität Hagen, Universitätsrechenzentrum. <ftp://ftp.fernuni-hagen.de/pub/pdf/urz-broschueren/broschueren/a0260003.pdf>.

- Kapr, A. (1963). *Buchgestaltung*. Dresden: VEB Verlag der Kunst.
- Kapr, A. (1975). *Hundertein Sätze zur Buchgestaltung*. Leipzig: Deutsche Bäckerei Leipzig.
- Klöckl, I. (2002). *L^AT_EX – Tipps & Tricks: Layoutanpassungen, Programmierung, Grafik, Hilfsprogramme, Zeichensätze* (2 ed.). Heidelberg: dpunkt Verlag.
- Knappen, J. (2004). *Schnell ans Ziel mit L^AT_EX 2_ε* (2 ed.). München: Oldenbourg Verlag.
- Knuth, D. E. (1979). *T_EX and METAFONT—New Directions in Typesetting*. Bedford, Massachusetts: American Mathematical Society and Digital Press.
- Knuth, D. E. (1981). *The Art of Computer Programming, Seminumerical Algorithms* (2 ed.), Volume 1. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Knuth, D. E. (1986a). *Computer Modern Typefaces*, Volume E, Computers and Typesetting. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Knuth, D. E. (1986b). *METAFONT The Program*, Volume D, Computers and Typesetting. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Knuth, D. E. (1986c). *The METAFONTbook*, Volume C, Computers and Typesetting. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Knuth, D. E. (1986d). *T_EX The Program*, Volume B, Computers and Typesetting. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Knuth, D. E. (1986e). *The T_EXbook*, Volume A, Computers and Typesetting. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Kohm, M. and J.-U. Morawski (2003). *KOMA-Script – Eine Sammlung von Klassen und Paketen für L^AT_EX 2_ε*. Heidelberg: DANTE e.V., Lehmanns Fachbuchhandlung. Anleitung zur Version 2.9 o; PDF-Version <http://www.ctan.org/tex-archive/doc/latex/koma-script/scrguide.pdf>.
- Kopka, H. (1996). *L^AT_EX, Band 3: Erweiterungen*. Bonn: Addison-Wesley Longman Verlag.
- Kopka, H. (1997). *L^AT_EX, Band 2: Ergänzungen* (2 ed.). Bonn: Addison-Wesley Longman Verlag.
- Kopka, H. (2000). *L^AT_EX, Band 1: Einführung* (2 ed.). Bonn: Addison-Wesley Longman Verlag.

Kopka, H. and P. W. Daly (1995). *A Guide to L^AT_EX 2_ε—Document Preparation for Beginners and Advanced Users* (2 ed.). Wokingham, England: Addison-Wesley Publishing Company. Übersetzung von L^AT_EX Band 1: Einführung, mit Anpassungen für den englischsprachigen Raum.

Lamport, L. (1994). *L^AT_EX—A Document Preparation System*. Reading, Massachusetts: Addison-Wesley Publishing Company.

Lamport, L. (1995). *Das L^AT_EX Handbuch*. Bonn: Addison-Wesley Verlag. Übersetzung von L^AT_EX—A Document Preparation System.

L^AT_EX3 Projekt Team (2001). *L^AT_EX 2_ε for Authors*. <http://www.ctan.org/tex-archiv/macros/latex/doc/usrguide.pdf>.

L^AT_EX3 Projekt Team (2003a). *Configuration options for L^AT_EX 2_ε*. <http://www.ctan.org/tex-archive/macros/latex/doc/cfgguide.pdf>.

L^AT_EX3 Projekt Team (2003b). *L^AT_EX 2_ε for Class and Package Writers*. <http://www.ctan.org/tex-archive/macros/latex/doc/clsguide.pdf>.

L^AT_EX3 Projekt Team (2004). *L^AT_EX 2_ε font selection*. <http://www.ctan.org/tex-archive/macros/latex/doc/fntguide.pdf>.

Love, T. (1999). *Advanced L^AT_EX*. http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/latex_advanced/latex_advanced.html.

Love, T. (2004). *L^AT_EX Math and Graphics*. http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/latex_maths+pix/latex_maths+pix.html.

L^yX Team (1997). *L^yX Reference Manual*. Teil der Online-Dokumentation.

L^yX Team (2003a). *Das L^yX-Tutorium*. Teil der Online-Dokumentation.

L^yX Team (2003b). *Einführung in L^yX oder Das Einmaleins der L^yX-Dokumentation*. Teil der Online-Dokumentation.

L^yX Team (2004a). *Das L^yX-Benutzerhandbuch*. Teil der Online-Dokumentation.

L^yX Team (2004b). *Profi-Tipps: erweiterte Textbearbeitung mit L^yX*. Teil der Online-Dokumentation.

Martinsen, T., S. Gilmore, and G. D. Greenwade (1996). *L^AT_EX 2_ε Help*. (Version 1.6). <http://www.ctan.org/tex-archive/info/latex2e-help-texinfo/latex2e.html>.

Niedermair, E. and M. Niedermair (2004). *L^AT_EX Praxisbuch*. Professional Series. Pöng: Franzis Verlag. Ungekürzte Originalausgabe zum DANTE-Jubiläum.

Partl, H. and A. Kielhorn (1996). *Layout-Änderungen mit L^AT_EX*. <http://www.ctan.org/tex-archive/info/lshort/german/>.

Perrousseaux, Y. (1995). *Manuel de Typographie Française élémentaire* (3 ed.). Reillanne: Atelier Perrousseaux.

Raichle, B. (1998). *Kurzbeschreibung german.sty und ngerman.sty*. (Version 2.5). Dokumentation zu den Paketen german und ngerman.

Raichle, B., R. Niepraschk, and T. Hafner (2003). *Fragen und Antworten (FAQ) über das Textsatzsystem T_EX und DANTE, Deutschsprachige Anwendervereinigung T_EX e. V. DANTE e. V. (Version 72)*. <http://www.dante.de/faq/de-tex-faq/>.

Rechenberg, P. (2002). *Technisches Schreiben*. München: Carl Hanser Verlag.

Salberg-Steinhardt, B. (1983). *Die Schrift, Geschichte – Gestaltung – Anwendung*. Köln: Dumont.

Schmidt, W. (2004). *Using common PostScript fonts with L^AT_EX*. Benutzerhandbuch zum Paket psnfss.

Schmidt, W., J. Knappen, H. Partl, and I. Hyna (1999). *L^AT_EX₂ ϵ -Kurzbeschreibung*. (Version 2.1). <http://www.ctan.org/tex-archive/info/lshort/german/>.

Schwarz, N. (1991). *Einführung in T_EX; inkl. Version 3.0* (3 ed.). Bonn: DANTE e. V., Lehmanns Fachbuchhandlung. PDF-Version <http://www.ruhr.uni-bochum.de/www-rz/schwanbs/TeX/>.

Spierkermann, E. (1994). *Ursache & Wirkung, Ein Typographischer Roman*. Mainz: Verlag Hermann Schmidt. Faksimilierter Nachdruck der Originalausgabe von 1986 mit einem Nachtrag des Verfassers, die technische Entwicklung seitdem betreffend.

Tschichold, J. (1987). *Ausgewählte Aufsätze über Fragen der Gestalt des Buches und der Typographie* (2 ed.). Basel: Birkhäuser.

Tschichold, J. (1992). *Erfreuliche Drucksachen durch gute Typographie – Eine Fibel für jedermann* (2 ed.). Augsburg: Maro Verlag.

Tschichold, J. (1995). *The new Typography*. Berkeley: University of California Press. Übersetzung von *Die Neue Typographie – Ein Handbuch für Zeitgemäß Schaffende*, 1987.

Voss, H. (2005). *Math Mode*. (Version 2.09). <http://www.ctan.org/tex-archive/info/math/voss/mathmode/Mathmode.pdf>.

Willberg, H. P. and F. Forssman (1997). *Lesetypographie*. Mainz: Verlag Hermann Schmidt.

Willberg, H. P. and F. Forssman (1999). *Erste Hilfe in Typographie – Ratgeber für Gestaltung mit Schrift*. Mainz: Verlag Hermann Schmidt.

Beispielindex

Buch1/AmSLayout/

amsref-cite.tex, 475
amsref-ltb.tex, 472
amsthmex.tex, 438

Buch1/AmSPakete/

align1.tex, 386
align1notag.tex, 387
alignat1.tex, 388
alignat2.tex, 388
aligned.tex, 394
alignedat1.tex, 395
alignedat2.tex, 395
amsbsyxmpl.tex, 423
amscd.tex, 421
amsmath.tex, 372
arrows.tex, 413
binom.tex, 405
binomex.tex, 405
boxed.tex, 411
cases.tex, 392
cfrac.tex, 404
dots.tex, 418
flalign1.tex, 389
flalign2.tex, 389
frac1.tex, 403
frac2.tex, 403
gather.tex, 385
gathered.tex, 393
genfrac.tex, 408
genfrac2.tex, 408
hdotsfor.tex, 400
iiint.tex, 409
iiintd.tex, 409

matrix1.tex, 397
matrix2.tex, 398
matrix3.tex, 399
mod.tex, 410
multiline1.tex, 383
multiline2.tex, 383
multiline3.tex, 383
multiline4.tex, 384
nobreakdash.tex, 375
numberwithin.tex, 376
raisetag.tex, 377
roots.tex, 401
scomb.tex, 415
smash.tex, 412
split.tex, 393
split1.tex, 390
split2.tex, 391
split3.tex, 391
subequations.tex, 379
tag.tex, 374
text.tex, 425

Buch1/Babel/

abkuerz.tex, 211
babel.tex, 228
ngerman.tex, 215

Buch1/Basispakete/

alltt.tex, 172
graphpap.tex, 180
ifthen1.tex, 182
makeidx.tex, 191
shortvrb.tex, 196
showidx.tex, 193
textcomp-bsp.tex, 198

- textcomp-od.tex, 198
- whiledo.tex, 183
- Buch1/DokStrukt/
 - maketitle.tex, 58
 - section.tex, 60
 - ssubpara.tex, 67
 - tcont.tex, 61
 - tcont2.tex, 62
 - title.tex, 56
- Buch1/EigTextStrukt/
 - acc1.tex, 120
 - acc2.tex, 120
 - acc3.tex, 121
 - bfseries.tex, 116
 - bfseriesum.tex, 117
 - counter.tex, 135
 - counter2.tex, 135
 - ebox.tex, 125
 - efbox.tex, 126
 - efboxparm.tex, 127
 - emakebox.tex, 126
 - emtext.tex, 119
 - eoverfbox.tex, 126
 - epbox.tex, 129
 - erbox.tex, 128
 - erbox2.tex, 128
 - erbox3.tex, 130
 - esavebox.tex, 128
 - fill.tex, 133
 - laengen.tex, 134
 - large.tex, 118
 - mbox.tex, 124
 - schkombi.tex, 119
 - strich.tex, 122
 - strich2.tex, 123
 - untext.tex, 120
- Buch1/Ergaenz/
 - arraybd-pkg.tex, 232
 - arraybd-std.tex, 231
 - arrayc.tex, 234
 - arrayh.tex, 238
 - arrayh2.tex, 239
 - arrayp.tex, 237
 - arrayp2.tex, 237
 - bmxml1.tex, 241
 - bmxml2.tex, 242
 - bmxml3.tex, 243
 - dcolum.tex, 250
 - dcolum2.tex, 251
 - delarray1.tex, 252
 - delarray2.tex, 252
 - delarray3.tex, 253
 - enumext1.tex, 254
 - enumext2.tex, 254
 - fontbsp.tex, 256
 - hhline1.tex, 260
 - hhline2.tex, 261
 - hhline3.tex, 262
 - indentfirst.tex, 264
 - indentfirst0.tex, 264
 - multicols1.tex, 278
 - multicols2.tex, 280
 - tabularxa.tex, 291
 - tabularxabs.tex, 293
 - tabularxb.tex, 291
 - tabularxd.tex, 294
 - theorem.tex, 299
 - verbatim.tex, 310
 - verbatim2.tex, 312
 - xspace.tex, 315
- Buch1/ErsteSchritte/
 - bfquote.tex, 25
 - descr.tex, 41
 - dklassen.tex, 34
 - enum.tex, 40
 - ErstesBeispiel.tex, 13
 - gruppe.tex, 26
 - item.tex, 40
 - kommentar.tex, 27
 - kommentar2.tex, 27
 - latexlogo.tex, 21
 - latexlogokorr.tex, 21
 - math1.tex, 45
 - math2.tex, 46
 - math3.tex, 46
 - newcommand.tex, 22
 - newcommand2.tex, 22
 - quote.tex, 24
 - rule.tex, 20
 - section.tex, 38
 - tabb1.tex, 42
 - tabb2.tex, 42
 - tabb3.tex, 43
 - tabul.tex, 43
 - tabulquote.tex, 44
 - tcont.tex, 39
 - umlaut.tex, 30
 - umlautger.tex, 31

- umlautinp.tex, 31
- xxquote.tex, 25
- Buch1/Grafik/
 - colorbox.tex, 327
 - foto.tex, 342
 - graycolor.tex, 325
 - pagecolor.tex, 326
 - RotXY.tex, 331
 - spreiz.tex, 333
 - spreiz2.tex, 334
- Buch1/MathForm/
 - akzent.tex, 168
 - array.tex, 154
 - bigop.tex, 162
 - brace1.tex, 146
 - brace1a.tex, 147
 - brace1b.tex, 148
 - brace2.tex, 149
 - cal.tex, 156
 - displaymath.tex, 140
 - dispmath-fl.tex, 141
 - dots.tex, 166
 - eqnarray-l.tex, 143
 - eqnarray-lf.tex, 144
 - eqnarray-n.tex, 143
 - eqnarray.tex, 142
 - equation.tex, 141
 - expind.tex, 146
 - frac.tex, 151
 - math.tex, 139
 - not.tex, 160
 - plainops.tex, 158
 - relationen.tex, 161
 - sqrt.tex, 150
 - vbraces.tex, 165
- Buch1/PSnfss/
 - dingautolist.tex, 361
 - dinglines.tex, 361
 - dinglist.tex, 360
 - mathpazo.tex, 356
 - mathptmx.tex, 358
 - piautolist.tex, 364
 - pilist.tex, 363
 - pisonst.tex, 364
- Buch1/Sample2e/
 - sample2e.tex, 587
- Buch1/SpezTextStruk/
 - center.tex, 95
 - enumkomp.tex, 72
 - footnote1.tex, 99
 - footnote2.tex, 99
 - footnote3.tex, 100
 - footnote4.tex, 101
 - itemkomp.tex, 73
 - listen.tex, 76
 - margin1.tex, 101
 - margin2.tex, 102
 - quer.tex, 109
 - quotation.tex, 97
 - quote.tex, 96
 - ragged.tex, 95
 - tabb1.tex, 78
 - tabb2.tex, 79
 - tabb3.tex, 80
 - tabb4.tex, 80
 - tabb5.tex, 81
 - tabb6.tex, 81
 - tabb7.tex, 82
 - tabb8.tex, 94
 - tabular1.tex, 85
 - tabular2.tex, 88
 - tabular3.tex, 90
 - tabular4.tex, 93
 - trivlist.tex, 77
 - verse.tex, 98
- Buch1/Vorwort/
 - bsp.tex, viii
 - bspex.tex, ix
 - bspexq.tex, viii

Befehlsindex

`\'`, 81, 82
`\(`, 45, 143, 150, 186, 373, 417
`\)`, 45, 143, 150, 186, 373, 417
`\+`, 80, 81
`\,`, 419
`\-`, 35, 80, 123, 211
`\.`, 447
`\/`, 314
`\:`, 419
`\;`, 419
`\=`, 41, 78, 81, 82, 447
`?\'`, 121
`\[`, 45, 425
`\#`, 20, 30
`\$`, 30
`\%`, 30
`\&`, 30
```, 120  
`ˆ`, 120  
`Å`, 121  
`Æ`, 121  
`\@addpunct`, 482  
`\@makecaption`, 107  
`\@startsection`, 63  
`--`, 122  
`-`, 122  
`"-`, 35  
`"=`, 122  
`"``, 32  
`"'`, 32  
`"~`, 122  
`"`, 31, 32, 211, 425  
`\"`, 447, 448  
`"-`, 211  
`"<`, 32, 211  
`"=`, 211  
`">`, 32, 211  
`"`, 211  
`"``, 32, 211  
`"'`, 32, 211  
`"~`, 211  
`\"a`, 425  
`ˆ`, 120  
`\^`, 30, 447  
`\``, 447  
`Ø`, 121  
`Œ`, 121  
`¶`, 121  
`|`, 163, 166  
`\'`, 447  
`§`, 121  
`\~`, 30, 447  
`\<Befehlsname>`, 20, 22, 288, 289  
`\<Befehl>`, 21–23  
`\<Gliederungsbefehl>`, 37, 59, 61  
`\<Gliederungsbefehl>*`, 59  
`\<Längenbefehl>`, 133  
`\<Name>`, 127  
`\<Umgebungsname>`, 312  
`\<Zähler>`, 134  
`\_`, 30, 330  
`å`, 121  
`æ`, 121

$\_$ , 120  
 $\_$ , 120  
 $\!$ , 419  
 $\!$ , 121  
 $\.$ , 120  
 $\dagger$ , 121  
 $\ddagger$ , 121  
 $\>$ , 41, 78, 81  
 $\_$ , 120  
 $\<$ , 81  
 $\text{œ}$ , 121  
 $\text{°}$ , 120  
 $\text{ß}$ , 121  
 $\grave{}$ , 120  
 $\backslash*$ , 380, 381  
 $\sim$ , 120  
 $\sim$ , 120  
 $\{$ , 30, 163  
 $\}$ , 30, 163  
 $\}$ , 45, 425  
 $\text{'}$ , 81, 82  
  
 $\text{'}$ , 82  
 $\text{=}$ , 82  
 $\text{'}$ , 82  
 $\text{above}$ , 407  
 $\text{abovewithdelims}$ , 407  
 $\text{abs}$ , 416, 417  
 $\text{abstract}$ , 56  
 $\text{abstractname}$ , 212, 217, 219  
 $\text{accentedsymbol}$ , 368, 426  
 $\text{acute}$ , 167  
 $\text{addcontentsline}$ , 68, 107, 108  
 $\text{addpenalty}$ , 174  
 $\text{addtocontents}$ , 69, 108  
 $\text{addtocounter}$ , 134, 246  
 $\text{addtolength}$ , 74, 133, 246, 272  
 $\text{AE}$ , 121, 225, 447  
 $\text{ae}$ , 121, 225, 447  
 $\text{afterpage}$ , 229, 230  
 $\text{aleph}$ , 167  
 $\text{allowdisplaybreaks}$ , 380, 381  
 $\text{Alph}$ , 254  
 $\text{alph}$ , 254  
 $\text{alpha}$ , 156  
  
 $\text{alsoname}$ , 212, 217, 219  
 $\text{amalg}$ , 157, 357  
 $\text{and}$ , 57  
 $\text{angle}$ , 167, 490  
 $\text{appendix}$ , 68  
 $\text{appendixname}$ , 212, 217, 219  
 $\text{approx}$ , 159  
 $\text{approxeq}$ , 488  
 $\text{arabic}$ , 254, 375  
 $\text{arccos}$ , 163  
 $\text{arcsin}$ , 163  
 $\text{arctan}$ , 163  
 $\text{arg}$ , 163  
 $\text{arraybackslash}$ , 292, 293  
 $\text{arraycolsep}$ , 92  
 $\text{arrayrulewidth}$ , 231, 232, 260, 261, 263  
 $\text{arraystretch}$ , 92  
 $\text{ast}$ , 157  
 $\text{asympt}$ , 159  
 $\text{at}$ , 225  
 $\text{atop}$ , 158, 407  
 $\text{atopwithdelims}$ , 407  
 $\text{author}$ , 57, 58  
 $\text{AutoSpaceBeforeFDP}$ , 222  
  
 $\text{b}$ , 447  
 $\text{backepsilon}$ , 488  
 $\text{backmatter}$ , 67  
 $\text{backprime}$ , 490  
 $\text{backsim}$ , 488  
 $\text{backsimeq}$ , 488  
 $\text{backslash}$ , 163, 167  
 $\text{bar}$ , 167, 168, 289  
 $\text{barwedge}$ , 489  
 $\text{baselineskip}$ , 92, 131, 285  
 $\text{baselinestretch}$ , 131  
 $\text{Bbb}$ , 484, 485  
 $\text{Bbbk}$ , 490  
 $\text{because}$ , 488  
 $\text{Bef}$ , 192  
 $\text{begin}$ , 309, 312, 434  
 $\text{begin}\{[p|b|B|v|V]\text{matrix}\}$ , 396  
 $\text{begin}\{\text{abstract}\}$ , 55  
 $\text{begin}\{\text{alignat}\}$ , 387  
 $\text{begin}\{\text{alignedat}\}$ , 394  
 $\text{begin}\{\text{aligned}\}$ , 394

- `\begin{align}`, 386
- `\begin{alltt}`, 171
- `\begin{array}`, 152, 251
- `\begin{bibchapter}`, 450
- `\begin{bibdiv}`, 449
- `\begin{biblist*}`, 450, 451
- `\begin{biblist}`, 450, 451
- `\begin{bibsection}`, 449
- `\begin{cases}`, 392
- `\begin{CD}`, 420
- `\begin{center}`, 95
- `\begin{comment}`, 311
- `\begin{description}`, 41
- `\begin{dingautolist}`, 360
- `\begin{dinglist}`, 360
- `\begin{displaymath}`, 45, 140
- `\begin{document}`, 14, 28, 47, 54, 181, 190, 195, 266, 292, 295, 441, 448, 508, 577
- `\begin{enumerate}`, 40, 71, 253
- `\begin{eqnarray*}`, 142
- `\begin{eqnarray}`, viii, 142
- `\begin{equation}`, 141, 378
- `\begin{figure}`, 103
- `\begin{flalign}`, 388
- `\begin{gathered}`, 392
- `\begin{gather}`, 378, 385
- `\begin{itemize}`, 40, 72
- `\begin{landscape}`, 345
- `\begin{list}`, 74
- `\begin{longtable}`, 268, 270, 272
- `\begin{math}`, 45, 139
- `\begin{matrix}`, 398
- `\begin{minipage}`, 129
- `\begin{multicols*}`, 275
- `\begin{multicols}`, 275
- `\begin{multline}`, 378, 382
- `\begin{otherlanguage*}`, 209
- `\begin{otherlanguage}`, 209
- `\begin{Piautolist}`, 363
- `\begin{Pilist}`, 363
- `\begin{proof}`, 435
- `\begin{quotation}`, 96
- `\begin{quote}`, 96
- `\begin{smallmatrix}`, 397
- `\begin{split}`, 390
- `\begin{subarray}`, 415
- `\begin{subequations}`, 378
- `\begin{table}`, 103
- `\begin{tabular*}`, 83
- `\begin{tabularx}`, 290
- `\begin{tabular}`, viii, 43, 82
- `\begin{titlepage}`, 56
- `\begin{verbatim}`, 311
- `\begin{verse}`, 97
- `\begin{<Umgebungsname>}`, 23
- `\beta`, 156
- `\beth`, 490
- `\between`, 488
- `\bfseries`, 23, 64, 66, 85, 234, 296, 297, 437, 461
- `\bib`, 449, 450, 452, 454, 455, 461, 464, 465
- `\bib*`, 455, 461
- `\BibField`, 482
- `\bibitem`, 286, 287
- `\bibliography`, 448, 454
- `\bibliographystyle`, 448
- `\bibname`, 212, 217, 219, 450
- `\bibquotes`, 445, 476, 478
- `\bibselect`, 453, 454, 473
- `\bibselect*`, 454, 473
- `\BibSpec`, 476–478
- `\BibSpecAlias`, 477
- `\BibSpecs`, 464
- `\big...`, 406
- `\bigcap`, 162
- `\bigcirc`, 157
- `\bigcup`, 162
- `\Biggl`, 416
- `\biggl`, 416
- `\Biggr`, 416
- `\biggr`, 416
- `\Bigl`, 416
- `\bigl`, 416
- `\bigodot`, 162
- `\bigoplus`, 162
- `\bigotimes`, 162
- `\Bigr`, 416

- `\bigr`, 416
- `\bigskip`, 57, 132
- `\bigskipamount`, 272
- `\bigsqcup`, 162
- `\bigstar`, 490
- `\bigtriangledown`, 157
- `\bigtriangleup`, 157, 158
- `\biguplus`, 162
- `\bigvee`, 162
- `\bigwedge`, 162
- `\binom`, 404–407
- `\blacklozenge`, 490
- `\blacksquare`, 490
- `\blacktriangle`, 490
- `\blacktriangledown`, 490
- `\blacktriangleleft`, 488
- `\blacktriangleright`, 488
- `\bm`, 240–245
- `\bmdefine`, 244
- `\bmmax`, 245, 614
- `\bmod`, 410
- `\Bodytext`, 266
- `\boi`, 225
- `\bold`, 245, 485
- `\boldmath`, 242, 243, 354, 423
- `\boldsymbol`, 243, 421–423, 485
- `\boldsymbol`, 422
- `\boolean`, 184
- `\bot`, 167
- `\bowtie`, 159
- `\Box`, 167, 189
- `\boxdot`, 489
- `\boxed`, 410, 411
- `\boxminus`, 489
- `\boxplus`, 489
- `\boxtimes`, 489
- `\breve`, 167
- `\bsc{Knuth}`, 223
- `\bullet`, 157
- `\Bumpeq`, 488
- `\bumpeq`, 488
- `\bysame`, 478, 480, 481
- 
- `\c`, 447
- `\cal`, 156
- `\Cap`, 489
- `\cap`, 157
- 
- `\caption`, 106, 107, 267, 269–272, 288
- `\caption*`, 270
- `\CardinalNumber`, 462
- `\CardinalNumeric`, 479, 482
- `\ccname`, 212, 217, 219
- `\cdot`, 157, 250
- `\cdots`, 417, 418
- `\centerdot`, 489
- `\centering`, 95, 240, 269, 292, 293
- `\cfrac`, 404
- `\chapter`, viii, 37, 52, 59–63, 68, 450
- `\chapter*`, 60
- `\chaptername`, 212, 217, 219
- `\check`, 167
- `\checkmark`, 490
- `\chi`, 156
- `\circ`, 157
- `\circeq`, 488
- `\circle`, 179
- `\circlearrowleft`, 487
- `\circlearrowright`, 487
- `\circledast`, 489
- `\circledcirc`, 489
- `\circleddash`, 489
- `\circledR`, 490
- `\circledS`, 490
- `\circonflexe`, 225
- `\cite`, 286, 287, 472–474, 530, 540
- `\cite{<Schl>}`, 473
- `\citelist`, 443, 473
- `\cites`, 443, 473
- `\ck`, 214
- `\cleardoublepage`, 104
- `\clearpage`, 104, 230
- `\cline`, 87–89, 91, 153, 258, 259, 269
- `\clubsuit`, 167
- `\cmd`, 245
- `\color`, 325, 326
- `\colorbox`, 326, 327
- `\columnbreak`, 282
- `\columnsep`, 51, 283
- `\columnseprule`, 51, 276, 283
- `\complement`, 490

- `\cong`, 159
- `\contentsname`, 212, 217, 219
- `\coprod`, 162, 357
- `\copyright`, 197, 255
- `\cos`, 163
- `\cosh`, 163
- `\cot`, 163
- `\coth`, 163
- `\countdown`, 182
- `\csc`, 163
- `\Cup`, 489
- `\cup`, 157
- `\curlyeqprec`, 488
- `\curlyeqsucc`, 488
- `\curlyvee`, 489
- `\curlywedge`, 489
- `\CurrentOption`, 289
- `\curvearrowleft`, 487
- `\curvearrowright`, 487
  
- `\d`, 447
- `\dagger`, 157
- `\daleth`, 490
- `\dashleftarrow`, 487
- `\DashPages`, 482
- `\dashrightarrow`, 487
- `\dashv`, 159
- `\date`, 57, 58, 210
- `\dateaustralian`, 216
- `\dateaustrian`, 213
- `\datebritish`, 216
- `\dateenglish`, 216
- `\datefrench`, 220
- `\dategerman`, 213
- `\dbinom`, 404–406
- `\ddagger`, 157
- `\ddddot`, 417
- `\dddot`, 417
- `\ddot`, 167, 417
- `\decimalsep`, 224
- `\DeclareBoldMathCommand`, 244
- `\DeclareGraphicsExtensions`, 343
- `\DeclareGraphicsRule`, 338, 339, 343, 344
- `\DeclareMathOperator`, 368, 424, 425
- `\DeclareMathOperator*`, 424, 425
- `\DeclareNameAccent`, 447, 448
- `\DeclareNameSymbol`, 448
- `\DeclareOption`, 289
- `\DeclareOption*`, 289
- `\def`, 288
- `\DefineAdditiveKey`, 477, 478
- `\definecolor`, 287, 323, 325
- `\DefineDummyKey`, 477
- `\DefineJournal`, 444
- `\DefinePublisher`, 444
- `\DefineSimpleKey`, 477
- `\DefineSupersedingKey`, 477
- `\deg`, 163
- `\degre`, 223
- `\degres`, 223
- `\degres{C}`, 223
- `\Delta`, 156
- `\delta`, 156
- `\depth`, 333, 337, 338
- `\depthof`, 247
- `\det`, 163
- `\dfrac`, 402, 406
- `\DH`, 447
- `\dh`, 447
- `\diagdown`, 490
- `\diagup`, 490
- `\Diamond`, 167, 189
- `\diamond`, 157
- `\diamondsuit`, 167
- `\digamma`, 490
- `\dim`, 163
- `\dimen`, 174
- `\ding`, 359, 360, 362, 363
- `\dingfill`, 361, 364
- `\dingline`, 361
- `\dinglist`, 364
- `\displaybreak`, 381
- `\displaystyle`, 403, 407
- `\div`, 157
- `\divideontimes`, 489
- `\DJ`, 209, 447
- `\dj`, 209, 447
- `\dlc`, 263

`\documentclass`, 19, 20, 28, 33, 34, 47, 49, 54, 140, 141, 175, 256, 268, 292, 329, 370, 448, 577, 578  
`\documentstyle`, 577  
`\dot`, 167, 417  
`\Doteq`, 488  
`\doteq`, 159  
`\doteqdot`, 488  
`\dotfill`, 20, 21  
`\dotplus`, 489  
`\dots`, 165, 218, 400, 401  
`\doublebarwedge`, 489  
`\doublecap`, 489  
`\doublecup`, 489  
`\doublerulesep`, 84, 87, 92, 260, 262, 263  
`\Downarrow`, 159, 163  
`\downarrow`, 159, 163  
`\downdownarrows`, 487  
`\downharpoonleft`, 487  
`\downharpoonright`, 487  
`\dq`, 30, 32  
  
`\editiontext`, 462, 479, 481  
`\ell`, 167  
`\elvssf`, 194  
`\em`, 174  
`\emergencystretch`, 282, 283  
`\emminnershape`, 174  
`\emph`, 119, 174, 528  
`\emptyset`, 167  
`\enclname`, 212, 217, 219  
`\end`, 309–312, 434  
`\end{[p | b | B | v | V]matrix}`, 396  
`\end{abstract}`, 55  
`\end{alignedat}`, 394  
`\end{aligned}`, 394  
`\end{align}`, 386, 387  
`\end{alltt}`, 171  
`\end{array}`, 152, 251  
`\end{bibchapter}`, 450  
`\end{bibdiv}`, 449  
`\end{biblist*}`, 450, 451  
`\end{biblist}`, 450, 451  
`\end{bibsection}`, 449  
`\end{cases}`, 392  
`\end{CD}`, 420  
`\end{center}`, 95  
`\end{comment}`, 311  
`\end{description}`, 41  
`\end{dingautolist}`, 360  
`\end{dinglist}`, 360  
`\end{displaymath}`, 45, 140  
`\end{document}`, 14, 28, 54, 104, 448, 508  
`\end{enumerate}`, 40, 71, 253  
`\end{eqnarray*}`, 142  
`\end{eqnarray}`, viii, 142  
`\end{equation}`, 141, 378  
`\end{figure}`, 103  
`\end{flalign}`, 388  
`\end{gathered}`, 392  
`\end{gather}`, 378, 385  
`\end{itemize}`, 40, 72  
`\end{landscape}`, 345  
`\end{list}`, 74  
`\end{longtable}`, 268, 272  
`\end{math}`, 45, 139  
`\end{matrix}`, 398  
`\end{minipage}`, 129  
`\end{multicols}`, 275  
`\end{multicols*}`, 275  
`\end{multline}`, 378, 382  
`\end{otherlanguage*}`, 209  
`\end{otherlanguage}`, 209  
`\end{Piautolist}`, 363  
`\end{Pilist}`, 363  
`\end{proof}`, 435  
`\end{quotation}`, 96  
`\end{quote}`, 96  
`\end{smallmatrix}`, 397  
`\end{split}`, 390  
`\end{subarray}`, 415  
`\end{subequations}`, 378  
`\end{table}`, 103  
`\end{tabular*}`, 83  
`\end{tabularx}`, 290  
`\end{tabular}`, 43, 82  
`\end{titlepage}`, 56  
`\end{verbatim*}`, 310  
`\end{verbatim}`, 309–311  
`\end{verse}`, 97

- `\end{<Umgebungsname>}`,  
23
- `\end{<Umgebungsname>}`, 312
- `\ende`, 308
- `\endfirsthead`, 270
- `\endfoot`, 270
- `\endhead`, 270
- `\endlastfoot`, 270
- `\eprint`, 483
- `\eprintpages`, 483
- `\epsilon`, 156
- `\eqcirc`, 488
- `\eqref`, 374
- `\eqsim`, 488
- `\eqslantgtr`, 488
- `\eqslantless`, 488
- `\equal`, 184, 308
- `\equiv`, 159
- `\eta`, 156
- `\etalchar`, 481
- `\etaltext`, 481
- `\eth`, 490
- `\ExecuteOption`, 319
- `\exists`, 167
- `\exp`, 163
- `\externaldocument`, 313,  
314
- `\extracolsep`, 83, 87, 88, 90, 91, 234,  
235, 268, 272, 290
- `\extrarowheight`, 238
- `\extratabsurround`, 237
- `\fallingdotseq`, 488
- `\fbox`, 125, 127, 288, 326, 327, 331,  
340, 410
- `\fboxrule`, 127, 327, 411
- `\fboxsep`, 127, 327, 340, 411
- `\fcolorbox`, 326, 327
- `\fg`, 222
- `\figurename`, 212, 217, 219
- `\fill`, 87, 91, 132, 133, 272,  
290
- `\Finv`, 490
- `\firsthline`, 237
- `\firstmark`, 174
- `\flat`, 167
- `\flq`, 32, 209
- `\flqq`, 32, 209
- `\flushbottom`, 192
- `\flushcolumns`, 282
- `\fnsymbol`, 100, 174
- `\fontencoding`, 115, 178
- `\fontsample`, 255
- `\foo`, 289
- `\Footertext`, 266
- `\footnote`, 98–100, 129, 272,  
528
- `\footnotemark`, 99, 272
- `\footnoterule`, 256
- `\footnotesize`, 451
- `\footnotetext`, 99, 272
- `\forall`, 167
- `\foreignlanguage`, 208, 209
- `\fprimo`, 225
- `\fquarto`, 225
- `\frac`, 46, 151, 402, 404, 406,  
407
- `\fracwithdelims`, 368, 426
- `\frak`, 485, 491
- `\framebox`, 125, 127
- `\FrenchLabelItem`, 222
- `\FrenchEnumerate`, 225
- `\FrenchEnumerate{2}`, 225
- `\FrenchFootnotes`, 221
- `\FrenchGuillemetsFrom`,  
225
- `\FrenchLayout`, 221
- `\FrenchPopularEnumerate`,  
225
- `\frontmatter`, 67
- `\frown`, 159
- `\frq`, 32, 209
- `\frqq`, 32, 209
- `\fsecundo`, 225
- `\ftertio`, 225
- `\fullref`, 302
- `\fussy`, 36
- `\Game`, 490
- `\Gamma`, 156
- `\gamma`, 156
- `\gcd`, 163
- `\genfrac`, 406–408, 426
- `\genfrak`, 408
- `\geq`, 159
- `\geqq`, 488
- `\geqslant`, 488
- `\gg`, 159

`\ggg`, 488  
`\gggtr`, 488  
`\gimel`, 490  
`\glossaryname`, 212, 217, 219  
`\glq`, 32, 209  
`\glqq`, 32, 209  
`\gnapprox`, 489  
`\gneq`, 489  
`\gneqq`, 489  
`\gnsim`, 489  
`\graphicspath`, 342, 343  
`\graphpaper`, 179  
`\grave`, 167  
`\grq`, 32, 209  
`\grqq`, 32, 209  
`\gtrapprox`, 488  
`\gtrdot`, 489  
`\gtreqless`, 488  
`\gtreqqlless`, 488  
`\gtrless`, 488  
`\gtrsimeq`, 488  
`\guillemotleft`, 209  
`\guillemotright`, 209  
`\guilsinglleft`, 209  
`\guilsinglright`, 209  
`\gvertneqq`, 489  
  
`\H`, 447  
`\hat`, 167  
`\hbar`, 167, 490  
`\hbox`, 35, 36  
`\hdotsfor`, 399–401  
`\Headertext`, 266  
`\headtoname`, 212, 217, 219  
`\heartsuit`, 167  
`\heavymath`, 243  
`\heavysymbol`, 243  
`\height`, 333, 337, 338  
`\heightof`, 247  
`\hfill`, 133, 530  
`\hhline`, 258–261, 263  
`\hline`, 86–88, 91, 153, 237, 258, 261, 269, 361  
`\hm`, 240, 241, 243, 244  
`\hmdefine`, 244  
`\hmmax`, 245, 614  
`\hoffset`, 265  
`\hom`, 163  
  
`\hookleftarrow`, 159  
`\hookrightarrow`, 159  
`\hphantom`, 148  
`\hrule`, 274  
`\hspace`, 293  
`\hslash`, 490  
`\hspace`, 91, 132, 161, 234, 263, 375, 388, 419  
`\Huge`, 433  
`\huge`, 433  
`\hyphenation`, 36  
  
`\i`, 120, 447  
`\idotsint`, 409  
`\ieme`, 223  
`\iemes`, 223  
`\ier`, 220, 223  
`\iere`, 223  
`\ieres`, 223  
`\iers`, 223  
`\IfEmptyBibField`, 482  
`\ifthenelse`, 181, 182, 186, 308  
`\iiiint`, 409  
`\iiint`, 409  
`\iint`, 409  
`\IJ`, 209  
`\ij`, 209  
`\Im`, 167  
`\imath`, 167  
`\in`, 159  
`\include`, 313  
`\includegraphics`, 338, 340, 343, 344, 346  
`\includegraphics*`, 334  
`\index`, 190, 192, 441  
`\indexname`, 212, 217, 219  
`\inf`, 163  
`\infty`, 167, 425  
`\injl`, 424  
`\input`, 190  
`\inputencoding`, 187  
`\int`, 162, 409  
`\intercal`, 489  
`\intertext`, 381  
`\iota`, 156  
`\isodd`, 183, 184  
`\issuetext`, 481  
`\isundefined`, 184

- `\item`, 39–41, 71–77, 185, 253, 290, 360, 363
- `\itemindent`, 77
- `\itshape`, 25, 64, 66, 256, 295, 297, 437, 451, 464
- `\j`, 120, 447
- `\jmath`, 167, 357
- `\Join`, 159, 189
- `\k`, 447
- `\kappa`, 156
- `\ker`, 163
- `\kill`, 42, 79, 269
- `\L`, 447
- `\l`, 447
- `\label`, 109, 141, 142, 254, 271, 286–288, 302, 304, 308, 313, 314, 374, 540
- `\labelformat`, 304, 305
- `\labelwidth`, 77, 254
- `\Lambda`, 156
- `\lambda`, 156
- `\langle`, 163
- `\lara`, 439, 440
- `\LARGE`, 433
- `\Large`, 64, 433
- `\large`, 433
- `\larger`, 433
- `\lasthline`, 237
- `\lastline`, 237
- `\LasyGuillemets`, 224
- `\LaTeX`, 21
- `\layout`, 266
- `\layout*`, 266
- `\lceil`, 163
- `\ldots`, 417, 418
- `\leadsto`, 159, 189
- `\left`, 164, 165, 243, 398, 406, 416
- `\Leftarrow`, 159
- `\leftarrow`, 159
- `\leftarrowtail`, 487
- `\lefteqn`, 143
- `\leftharpoondown`, 159
- `\leftharpoonup`, 159
- `\leftleftarrows`, 487
- `\leftmargin`, 76, 77, 96
- `\Leftrightarrow`, 159
- `\leftrightarrow`, 159
- `\leftrightarrows`, 487
- `\leftrightharpoons`, 487
- `\leftrightsquigarrow`, 487
- `\leftroot`, 401
- `\leftthreetimes`, 489
- `\lengthtest`, 183
- `\leq`, 159
- `\leqq`, 488
- `\leqslant`, 488
- `\lessapprox`, 488
- `\lessdot`, 489
- `\lesseqgtr`, 488
- `\lesseqqgtr`, 488
- `\lessgtr`, 488
- `\lesssim`, 488
- `\lfloor`, 163
- `\lg`, 163
- `\lhd`, 157, 189
- `\Lim`, 425
- `\lim`, 163, 368
- `\liminf`, 163
- `\limits`, 409, 422, 424
- `\limsup`, 163
- `\linebreak`, 482
- `\linespread`, 350
- `\listfigurename`, 212, 217, 219
- `\listoffigures`, 107
- `\listoftables`, 107
- `\listtablename`, 212, 217, 219
- `\ll`, 159
- `\llcorner`, 490
- `\Lleftarrow`, 487
- `\lll`, 488
- `\llless`, 488
- `\ln`, 163
- `\lnapprox`, 489
- `\lneq`, 489
- `\lneqq`, 489
- `\lnsim`, 489
- `\log`, 163
- `\Longleftarrow`, 159
- `\longleftarrow`, 159
- `\Longleftrightarrow`, 159
- `\longleftrightarrow`, 159

- `\longmapsto`, 159
- `\Longrightarrow`, 159
- `\longrightarrow`, 159
- `\looparrowleft`, 487
- `\looparrowright`, 487
- `\lozenge`, 490
- `\lrcorner`, 490
- `\Lsh`, 487
- `\LTcapwidth`, 270, 272
- `\ltimes`, 489
- `\LTleft`, 268, 272
- `\LTpost`, 272
- `\LTpre`, 272
- `\LTRight`, 268, 272
- `\lVert`, 416
- `\lvert`, 416
- `\lvertneqq`, 489
- `\Längenbefehl`, 133
- 
- `\mainmatter`, 67
- `\makebox`, 124, 125, 127, 148, 246
- `\makeindex`, 190, 441
- `\maketitle`, 56–58
- `\maltese`, 490
- `\mapsto`, 159
- `\MarginNotestext`, 266
- `\marginpar`, 101, 102, 274, 528
- `\markboth`, 52, 53, 60
- `\markright`, 52, 53
- `\math`, 157
- `\math...`, 244
- `\mathb`, 485
- `\mathbb`, 484
- `\mathbf`, 156
- `\mathbf`, 355, 357
- `\mathbfbold`, 355
- `\mathcal`, 155, 356, 429, 492
- `\mathfrak`, 485, 491
- `\mathindent`, 51
- `\mathit`, 157
- `\mathit`, 156
- `\mathnormal`, 156
- `\mathnormal`, 156
- `\mathrel`, 245
- `\mathrm`, 156
- `\mathrm`, 241, 242, 355, 424
- `\mathscr`, 492
- 
- `\mathsf`, 157
- `\mathtt`, 157
- `\mathversion`, 242, 422, 485
- `\max`, 163
- `\mbox`, 123–125, 127, 128, 143, 149, 150, 242, 326, 435
- `\measuredangle`, 490
- `\medskip`, 132
- `\medspace`, 419
- `\medthickspace`, 419
- `\mho`, 167, 189, 490
- `\mid`, 159
- `\min`, 163
- `\mod`, 410
- `\models`, 159
- `\mp`, 157
- `\mspace`, 419
- `\mthversion`, 423
- `\mu`, 156
- `\multicolbaselineskip`, 283
- `\multicolovershoot`, 284
- `\multicolpretolerance`, 286
- `\multicolsep`, 283
- `\multicoltolerance`, 286
- `\multicolumn`, 86, 89, 90, 153, 263, 268–270, 294, 386, 387, 392, 396, 399
- `\multicolundershoot`, 284
- `\multimap`, 488
- `\multitup`, 179
- 
- `\nabla`, 167
- `\natural`, 167
- `\ncong`, 489
- `\nearrow`, 159
- `\neg`, 167
- `\negmedspace`, 419
- `\negthinspace`, 419
- `\neq`, 159
- `\newcolumn`, 251
- `\newcolumnntype`, 234–236, 290
- `\newcommand`, 22, 23, 25, 64, 235, 245, 288, 289, 293, 308, 375, 407, 416, 426
- `\newcommand{\bmmax}`, 245
- `\newcommand{\hmmax}`, 245

`\newcounter`, 76  
`\newenvironment`, 24  
`\newline`, 36, 437  
`\newpage`, 37, 69, 271  
`\newsavebox`, 127  
`\newtheorem`, 295, 296, 433, 434, 436, 437  
`\newtheorem*`, 434  
`\newtheoremstyle`, 437  
`\nexists`, 490  
`\NG`, 447  
`\ng`, 447  
`\ngeq`, 489  
`\ngeqq`, 489  
`\ngeqslant`, 489  
`\ngtr`, 489  
`\ni`, 159  
`\nLeftarrow`, 487  
`\nleftarrow`, 487  
`\nLeftrightarrow`, 487  
`\nleq`, 489  
`\nleqq`, 489  
`\nleqslant`, 489  
`\nless`, 489  
`\nmid`, 489  
`\No`, 223  
`\no`, 223  
`\NoAutoSpaceBeforeFDP`, 222  
`\nobreakdash`, 374  
`\nocite`, 540  
`\nocite{*}`, 454  
`\nolimits`, 414  
`\nombre`, 223  
`\nonumber`, 142, 143, 374, 385, 386, 388, 389  
`\nopagbreak`, 271  
`\nopagebreak`, 271  
`\nopunct`, 482  
`\norm`, 416, 417  
`\normal`, 245  
`\normalfont`, 64  
`\normalsize`, 432, 433, 451  
`\not`, 159–161  
`\notag`, 374, 385, 386, 388, 389  
`\notshown`, 266  
`\nparallel`, 489  
`\nprec`, 489  
`\npreceq`, 489  
`\nrightarrow`, 487  
`\nrightharpoonup`, 487  
`\nshortmid`, 489  
`\nshortparallel`, 489  
`\nsim`, 489  
`\nsubseteq`, 489  
`\nsubseteqq`, 489  
`\nsucc`, 489  
`\nsucceq`, 489  
`\nsupseteq`, 489  
`\nsupseteqq`, 489  
`\ntriangleleft`, 489  
`\ntrianglelefteq`, 489  
`\ntriangleright`, 489  
`\ntrianglerighteq`, 489  
`\nu`, 156  
`\numberline`, 69, 108  
`\numberwithin`, 375  
`\nVDash`, 489  
`\nVdash`, 489  
`\nvDash`, 489  
`\nvdash`, 489  
`\nwarrow`, 159  
`\O`, 121, 447  
`\o`, 121, 447  
`\ocite`, 473, 474  
`\odot`, 157  
`\OE`, 225, 447  
`\oe`, 225, 447  
`\og`, 222  
`\oint`, 162  
`\oldstylenums`, 198  
`\Omega`, 156  
`\omega`, 156  
`\ominus`, 157  
`\oneinchtext`, 266  
`\operatorname`, 414, 424, 425  
`\operatorname*`, 424, 425  
`\oplus`, 157  
`\ord`, 181  
`\oslash`, 157  
`\otimes`, 157  
`\ovalbox`, 128  
`\over`, 407  
`\overbrace`, 146–149

`\overleftarrow`, 412  
`\overleftrightarrow`, 412  
`\overline`, 167, 168  
`\overrightarrow`, 412  
`\overset`, 413–415  
`\overwithdelims`, 407  
  
`\pagebreak`, 271, 282, 284, 345, 381  
`\pagecolor`, 326  
`\pagename`, 212, 217, 219  
`\pagenumbering`, 53, 303  
`\pageref`, 109, 141, 184, 286–288, 301, 303, 307, 309, 313, 540  
`\pagestyle`, 35, 52  
`\par`, 28, 96, 97, 115, 132  
`\paragraph`, 37, 59, 60, 62, 185  
`\parallel`, 159  
`\parbox`, 129, 232, 246, 270, 272  
`\parenthesize`, 476, 478  
`\parindent`, 34, 76, 233, 247  
`\parsep`, 75  
`\parskip`, 74  
`\part`, 37, 59, 60, 62, 63  
`\partial`, 167  
`\partname`, 212, 217, 219, 220  
`\perp`, 159  
`\phantom`, 148  
`\Phi`, 156  
`\phi`, 156  
`\Pi`, 156  
`\pi`, 45, 156  
`\Pifill`, 364  
`\Pifont`, 364  
`\Piline`, 364  
`\Pisymbol`, 363  
`\pitchfork`, 488  
`\Plural`, 482  
`\pm`, 157  
`\pmb`, 422, 487  
`\pmod`, 410  
`\pod`, 410  
`\poptab`, 80  
`\poptabs`, 80  
`\postmulticols`, 283  
`\Pr`, 163  
  
`\prec`, 159  
`\precapprox`, 488  
`\preccurlyeq`, 488  
`\preceq`, 159  
`\precnapprox`, 489  
`\precneqq`, 489  
`\precnsim`, 489  
`\precsim`, 488  
`\prefacename`, 212, 217, 219  
`\premulticols`, 283  
`\pretolerance`, 286  
`\prime`, 167, 414, 440  
`\primo`, 225  
`\Print...`, 476  
`\PrintAuthors`, 478  
`\PrintBook`, 478  
`\PrintConference`, 478  
`\PrintConferenceDetails`, 478  
`\PrintContributions`, 478  
`\PrintDate`, 478, 479  
`\PrintDateB`, 479  
`\PrintDatePosted`, 479  
`\PrintDatePV`, 479  
`\PrintDOI`, 479  
`\PrintEdition`, 479  
`\PrintEditorsA`, 479  
`\PrintEditorsB`, 479  
`\PrintEditorsC`, 479  
`\Printindex`, 441  
`\printindex`, 442  
`\PrintISBNs`, 480  
`\PrintPartials`, 480  
`\PrintPrimary`, 480  
`\PrintReprint`, 480  
`\PrintReviews`, 480  
`\PrintSeries`, 482  
`\PrintThesisType`, 480  
`\PrintTranslation`, 480  
`\PrintTranslatorsA`, 480  
`\PrintTranslatorsB`, 480  
`\PrintTranslatorsC`, 481  
`\PrintYear`, 481  
`\ProcessOptions`, 289  
`\prod`, 162, 414  
`\projlim`, 424  
`\proofname`, 212, 217, 219, 435  
`\propto`, 159

- \protect, 69, 108, 230, 272
- \ProvidesDefinition, 288, 289
- \psfig, 345
- \Psi, 156
- \psi, 156
- \pushtab, 80
- \pushtabs, 80
- \put, 179
- \qed, 436
- \qedhere, 435
- \qq, 439, 440
- \qqquad, 387, 388, 419
- \quad, 395, 419
- \quarto, 225
- \quotedblbase, 209
- \quotesinglbase, 209
- \r, 447
- \raggedcolumns, 282
- \raggedleft, 240, 269, 292
- \raggedright, 240, 269, 292
- \raisebox, 128
- \raisetag, 376, 377, 380
- \rangle, 163
- \ratio, 247
- \rceil, 163
- \Re, 167
- \real, 246, 247
- \Ref, 305
- \ref, 109, 141, 254, 286, 287, 302, 304, 305, 307–309, 313, 314, 373, 374, 540
- \reflectbox, 333
- \refname, 212, 217, 219, 450
- \reftext, 308
- \reftextafter, 304
- \reftextbefore, 304
- \reftextcurrent, 307
- \reftextface, 304
- \reftextfaraway, 303
- \reftextvario, 304
- \reftwo, 308
- \relax, 88
- \renewcommand, 23, 26, 92, 100, 105, 131, 174, 222, 288, 293, 303, 435
- \renewenvironment, 26
- \RequirePackage, 175, 288, 289
- \RequirePackage{fix-cm}, 175
- \resetbiblist, 451
- \resetbiblist{999}, 451
- \resizebox\*, 333
- \restriction, 488
- \rfloor, 163
- \rhd, 157, 189
- \rho, 156
- \right, 164, 165, 243, 398, 406, 416
- \Rightarrow, 159
- \rightarrow, 159, 425
- \rightarrowtail, 487
- \rightharpoondown, 159
- \rightharpoonup, 159
- \rightleftarrows, 487
- \rightleftharpoons, 159, 487
- \rightmargin, 76, 77, 96
- \rightrightarrows, 487
- \rightsquigarrow, 487
- \rightthreetimes, 489
- \risingdotseq, 488
- \rm, 193
- \rmfamily, 351, 352, 358
- \Roman, 254
- \roman, 254
- \rotatebox, 330, 337
- \Rrightarrow, 487
- \Rsh, 487
- \rtimes, 489
- \rule, 20, 89, 105, 130, 147, 239, 340, 400, 401
- \rVert, 416
- \rvert, 416
- \sameauthors, 480, 481, 483
- \savebox, 127
- \sb, 171
- \sbox, 127
- \sc, 194
- \scalebox, 332, 333
- \scriptscriptstyle, 407
- \scriptsize, 147
- \scriptstyle, 407

- \scshape, 25, 26, 174, 194, 256
- \searrow, 159
- \sec, 163
- \secnumdepth, 66
- \section, 37, 52, 59, 60, 62–65, 68, 135, 184, 185, 435, 436, 450
- \secundo, 225
- \see, 191
- \seealso, 191
- \seename, 212, 217, 219
- \selectfont, 115
- \selectlanguage, 208, 209, 218
- \setcounter, 61, 62, 66, 100, 105, 134, 246, 451
- \setemergencystretch, 282, 283
- \setlength, 34, 74, 76, 81, 133, 174, 233, 246, 247, 272, 300
- \setlongtables, 267, 273
- \setminus, 157
- \settodepth, 247
- \settoheight, 247
- \settowidth, 247
- \sf, 193
- \sffamily, 296, 350, 351, 358, 364
- \sharp, 167
- \shoveleft, 383
- \shoveright, 383
- \showcols, 236
- \sideset, 414, 415
- \Sigma, 156
- \sigma, 156
- \sim, 159
- \simeq, 159
- \sin, 163, 368
- \SingularPlural, 482
- \sinh, 163
- \slc, 263
- \sloppy, 36, 37
- \slshape, 256, 295, 296, 325
- \SMALL, 433
- \Small, 433
- \small, 55, 433
- \smaller, 433
- \smallfrown, 488
- \smallsetminus, 489
- \smallskip, 105, 132
- \smallsmile, 488
- \smash, 151, 411
- \smash[b | t], 411
- \smile, 159
- \sp, 171
- \spadesuit, 167
- \spbreve, 426
- \spcheck, 426
- \spdddot, 426
- \spddot, 426
- \spdot, 426
- \sphen, 426
- \sphericalangle, 490
- \sptilde, 426
- \sqcap, 157
- \sqcup, 157
- \sqrt, 150, 242
- \sqrtsign, 242
- \sqsubset, 159, 189, 488
- \sqsubseteq, 159
- \sqsupset, 159, 189, 488
- \sqsupseteq, 159
- \square, 490
- \SS, 209, 447, 448
- \ss, 30, 425, 447
- \ss{ }, 30
- \ssubparagraph, 66
- \stackrel, 158, 160, 413
- \StandardFootnotes, 221
- \StandardLayout, 221
- \StandardMathComma, 224
- \star, 157
- \start, 308
- \string:, 219
- \string;, 219
- \string?, 219
- \string!, 219
- \SubEtil, 481
- \subparagraph, 37, 59, 60, 62, 66
- \subsection, 37, 52, 59, 60, 62, 135, 435, 436
- \Subset, 488
- \subset, 159
- \subseteq, 159
- \subsubseteq, 488
- \subsetneq, 489

- `\subsetneqq`, 489
- `\substack`, 414, 415
- `\subsubsection`, 37, 59, 60, 62
- `\succ`, 159
- `\succapprox`, 488
- `\succcurlyeq`, 488
- `\succeq`, 159
- `\succnapprox`, 489
- `\succneqq`, 489
- `\succnsim`, 489
- `\succsim`, 488
- `\sum`, 162, 414, 422
- `\sup`, 163
- `\suppressfloats`, 104, 176
- `\Supset`, 488
- `\supset`, 159
- `\supseteq`, 159
- `\supseteqq`, 488
- `\supsetneq`, 489
- `\supsetneqq`, 489
- `\surd`, 167
- `\SwapBreak`, 482
- `\swapnumbers`, 436
- `\swarrow`, 159
- `\symbol`, 112
- `\syntaxonly`, 196
- `\t`, 197, 447
- `\tabbingsep`, 81
- `\tabcolsep`, 84, 154, 232
- `\tablename`, 212, 217, 220
- `\tableofcontents`, 38, 68, 69, 107
- `\tabularnewline`, 240, 269
- `\tabularxcolumn`, 293
- `\tag`, 373, 374
- `\tag*`, 373
- `\tan`, 163, 425
- `\tanh`, 163
- `\tau`, 156
- `\tbinom`, 404–407
- `\tenrm`, 194, 286
- `\tensf`, 286
- `\tertio`, 225
- `\texbullet`, 255
- `\text`, 368, 425
- `\text...`, 115, 119
- `\textasciicircum`, 439
- `\textasciitilde`, 439
- `\textbackslash`, 30, 439
- `\textbar`, 30, 439
- `\textbf`, 156
- `\textbullet`, 439
- `\textcircled`, 197
- `\textcolor`, 325
- `\textcompwordmark`, 439
- `\textemdash`, 123, 439
- `\textendash`, 123, 439
- `\texteuro`, 197, 350
- `\textexclamdown`, 439
- `\textgreater`, 30
- `\textheight`, 52, 265
- `\textit`, 157, 476
- `\textless`, 30
- `\textperiodcentered`, 439
- `\textquestiondown`, 439
- `\textquotedblleft`, 121, 439
- `\textquotedblright`, 121, 439
- `\textquoteleft`, 121, 439
- `\textquoteright`, 121, 439
- `\textrm`, 147, 156
- `\textsc`, 528
- `\textsf`, 157
- `\textsl`, 325
- `\textstyle`, 403, 407
- `\texttt`, 157
- `\textvisiblespace`, 439
- `\textwd`, 247
- `\textwidth`, 52, 105, 183, 265, 272, 337
- `\tfrac`, 402, 406
- `\tg`, 425
- `\TH`, 447
- `\th`, 447
- `\thanks`, 57, 58, 221
- `\theequation`, 378
- `\thefigure`, 305
- `\thefootnote`, 100
- `\theorem`, 63
- `\theorembodyfont`, 296, 297
- `\theoremheaderfont`, 296, 297
- `\theorempostskipamount`, 297

- `\theorempreskipamount`, 297
- `\theoremstyle`, 295, 296, 436
- `\thepage`, 135
- `\theparagraph`, 66
- `\therefore`, 488
- `\thesubpararaph`, 66
- `\Theta`, 156
- `\theta`, 156
- `\thevpagerefnum`, 304
- `\thickapprox`, 488
- `\thicksim`, 488
- `\thickspace`, 419
- `\thinspace`, 419
- `\ThinSpaceInFrenchNumbers`, 224
- `\thousandsep`, 224
- `\tild`, 225
- `\tilde`, 167
- `\times`, 157
- `\Tiny`, 433
- `\tiny`, 433
- `\Titel`, 107
- `\title`, 57, 58
- `\tocdepth`, 66
- `\today`, 210, 213, 217, 220
- `\tolerance`, 286
- `\top`, 167
- `\topfigure`, 105
- `\topfraction`, 105
- `\topnumber`, 105
- `\topsep`, 74
- `\totalheight`, 333, 338
- `\tprime`, 440
- `\traceoff`, 300
- `\traceon`, 300
- `\tracingall`, 300
- `\tracingassigns`, 300
- `\tracinggroups`, 300
- `\tracingtabularx`, 292
- `\triangle`, 167
- `\triangledown`, 490
- `\triangleleft`, 157
- `\trianglelefteq`, 488
- `\triangleq`, 488
- `\triangleright`, 157
- `\trianglerighteq`, 488
- `\tsup`, 440
- `\tt`, 194
- `\ttfamily`, 194, 351, 352, 358
- `\twocolumn`, 174, 185, 276
- `\twoheadleftarrow`, 487
- `\twoheadrightarrow`, 487
- `\u`, 447
- `\ulcorner`, 490
- `\unboldmath`, 242
- `\underbrace`, 146, 148–150
- `\underleftarrow`, 412
- `\underlefterightarrow`, 412
- `\underline`, 119, 167, 168
- `\underrightarrow`, 412
- `\underset`, 413–415
- `\unitlength`, 179
- `\unlhd`, 157, 189
- `\unrhd`, 157, 189
- `\up{er}`, 223
- `\up{me}`, 223
- `\Uparrow`, 159, 163
- `\uparrow`, 159, 163
- `\Updownarrow`, 159, 163
- `\updownarrow`, 159, 163
- `\upharpoonleft`, 487
- `\upharpoonright`, 487
- `\uplus`, 157
- `\uproot`, 401
- `\upshape`, 296
- `\Upsilon`, 156
- `\upsilon`, 156
- `\upuparrows`, 487
- `\urcorner`, 490
- `\url`, 483
- `\UseAllDefinitions`, 289
- `\usebox`, 127
- `\usecounter`, 76
- `\UseDefinition`, 289
- `\usepackage`, ix, 31, 34, 54, 64, 114, 115, 170, 173–176, 178, 181, 187, 189, 190, 192–194, 196, 197, 200, 206, 240, 245, 255, 265, 267, 274, 286–289, 294, 300, 301, 310, 313, 314, 323, 328, 345, 350–354, 357–359, 372, 420, 421, 433, 439, 440, 442, 446, 448, 474, 484, 485, 490, 492, 495, 525, 578

`\UseSomeDefinitions`, 289  
`\v`, 447  
`\value`, 183  
`\varDelta`, 418  
`\varepsilon`, 156  
`\varGamma`, 418  
`\varinjlim`, 424  
`\varkappa`, 490  
`\varLambda`, 418  
`\varliminf`, 424  
`\varlimsup`, 424  
`\varnothing`, 490  
`\varOmega`, 418  
`\varPhi`, 418  
`\varphi`, 156  
`\varPi`, 418  
`\varpi`, 156  
`\varprojlim`, 424  
`\varpropto`, 488  
`\varPsi`, 418  
`\varrho`, 156  
`\varSigma`, 418  
`\varsigma`, 156  
`\varsubsetneq`, 489  
`\varsubsetneqq`, 489  
`\varsupsetneq`, 489  
`\varsupsetneqq`, 489  
`\varTheta`, 418  
`\vartheta`, 156  
`\vartriangle`, 488  
`\vartriangleleft`, 488  
`\vartriangleright`, 488  
`\varUpsilon`, 418  
`\varXi`, 418  
`\vbox`, 285  
`\Vdash`, 488  
`\vDash`, 488  
`\vdash`, 159  
`\vec`, 167  
`\vee`, 157  
`\veebar`, 489  
`\verb`, 194, 195, 291  
`\verb*`, 194, 195  
`\verbatim`, 312  
`\verbatimfile`, 540  
`\verbatiminput`, 311  
`\verbatiminput*`, 311  
`\vfill`, 133  
`\vline`, 84, 90, 91, 153, 235  
`\voffset`, 265  
`\volttext`, 481  
`\vpageref`, 305–308  
`\vpageref*`, 305  
`\vpagerefrange`, 306, 307  
`\vpagerefrange*`, 306, 307  
`\vphantom`, 147  
`\Vref`, 305  
`\vref`, 302, 305, 306  
`\vref*`, 302, 305  
`\vrefpagenum`, 308  
`\vrefrange`, 306–308  
`\vrefshowerrors`, 309  
`\vrefwarning`, 309  
`\vspace`, 132, 133, 174  
`\vspace*`, 132  
`\Vdash`, 488  
`\wedge`, 157  
`\whiledo`, 182  
`\widehat`, 167, 487  
`\widetilde`, 167, 487  
`\width`, 333  
`\widthof`, 247  
`\wp`, 167  
`\wr`, 157  
`\Xi`, 156  
`\xi`, 156  
`\xleftarrow`, 412  
`\xrightarrow`, 412  
`\xspace`, 314  
`\ycite`, 473  
`\yen`, 490  
`\zeta`, 156

---

## Autorenindex

- Abrahamsen, Per, 509  
American Mathematical Society, ,  
433  
Arbortext, Inc., 319  
  
Beebe, Nelson, 319  
Beeton, Barbara, 440  
Bezos, Javier, 417  
Blue Sky Research, Inc., 319, 429, 484,  
494  
Braams, Johannes, 170, 178, 190, 197,  
205, 265  
  
Carlisle, David, 173, 178, 181, 190,  
197, 229, 230, 240, 248, 251, 253,  
258, 263, 266, 286, 290, 313, 314,  
317, 346  
Clark, James, 319  
  
Downes, Michael J., 367, 369, 428, 439,  
442  
  
Ettrich, Matthias, 523  
  
Fischer, Lars Peter, 509  
  
Gaulle, Bernard, 228  
  
Hafner, Jim, 322  
  
Jackowski, Boguslaw, 177  
Jeffrey, Alan, 178, 187, 190, 197, 255,  
286, 288, 356  
Jensen, Frank, 246, 493  
  
Jones, David M., 369, 428, 441  
  
Kelly, B. Hamilton, 319  
Kinch, Richard, 319  
Knappen, Jörg, 175  
Knuth, Donald E., 4, 10, 177,  
367  
Kohm, Markus, 49  
Krawitz, Robert, 501  
  
Lamport, Dr. Leslie, 7, 176, 178,  
205  
Lamport, Leslie, 190  
  
Martillo, Joachim, 501  
Mattes, Eberhard, 319  
McPherson, Kent, 265  
MicroPress, Inc., 319  
Mittelbach, Frank, 8, 173, 176, 178,  
187, 189, 190, 193, 194, 196, 197,  
200, 230, 240, 256, 273, 294, 300,  
301, 433, 493  
  
Partl, Dr. Hubert, 31  
Personal TeX, Inc., 319  
Puga, Diego, 354  
  
Raichle, Bernd, 31, 309  
Rathz, Sebastian, 356  
Rokicki, Tom, 319  
Rowley, Chris, 8, 173, 176, 178, 190,  
197, 246, 309  
Ryćko, M., 177

Schmidt, Walter, 173, 347  
Schöpf, Dr. Rainer, 8, 173, 178, 190,  
196, 197, 200, 309  
Scientific Word/Workplace, Inc.,  
319  
Sendoukas, Hippocrates, 319  
Spivak, Michael, 367  
Stallman, Richard, 501  
  
Thành, Hàn Thê, 319  
Thorup, Kresten Krab, 246,  
509  
Trevorrow, Andrew, 319

Umeki, Hideo, 265  
  
van der Hoeven, Joris, 533  
Van Zandt, Timothy, 318, 346  
Vieth, Ulrik, 356  
  
Williams, Graham, 12  
  
Y&Y, Inc., 319, 429, 484, 494  
  
Zapf, Herrmann, 351, 429, 483, 484,  
490–494

---

# Index

- abel, 450
- Abkürzungen, 211, 547
- Absatzabstand, 131
- Absatzboxen, 129
- Absatzeinrückung, 34
- Abstand, 131, 567
- Abstract, 55
- abstract, 55, 56, 58
- accents, 417
- Acrobat Reader, 18
- afterpage, 229
- Akronyme, 547
- Akut, 32
- Akzente, 120
- align, 371, 372, 386–390, 393, 394
- align\*, 371
- alignat, 371, 372, 387, 388, 394
- alignat\*, 371, 388
- aligned, 371, 389, 393–395
- alignedat, 371, 389, 394, 395
- alltt, 170, 171
- allt, 55, 170
- American Mathematical Society, 155
- $\mathcal{AMS}$ , 155
- amsart, 376, 430, 431, 433, 440
- amsbook, 376, 430, 431, 433, 440
- amsbsy, 243, 368, 369, 421, 430, 485, 487
- amscd, 368, 420
- amstdx, 430
- amsmath, 163, 167, 176, 189, 429, 430, 432, 483, 484, 487, 490, 492, 495
- amsgen, 421, 430
- amsjpa, 446
- amslatex, 430
- amsmath, 138, 150, 151, 158, 166, 176, 369, 372–377, 380, 381, 389, 391, 396, 399, 401–404, 406–408, 410–414, 416–421, 423–425, 428, 430, 431, 434, 435, 485, 530
- amsmidx, 430, 440, 441
- amsopn, 157, 368, 369, 423, 430
- amsproc, 376, 430
- amsrefs, 451
- amsrefs, 428, 430, 442, 444–453, 455–459, 461, 463–465, 467–469, 472–474, 476, 477
- amsrefs, 459
- amsrep, 433
- amssymb, 155, 157, 158, 166, 189, 429, 483–485, 487–490
- amstex, 5, 55, 374, 376, 426, 485
- amstext, 368, 369, 425, 430, 570
- amsthm, 63, 294, 430, 433, 435
- amxtra, 426
- Anführungsstriche, 32
- Anführungszeichen, 121, 550, 551

- doppelt, 32
- einfach, 32
- Anhang, 68
- Anwendervereinigung, 10
  - DANTE e. V., 11
  - Gutenberg, 10
  - NTG, 10
  - TUG, 10
- Apostroph, 32
- Argumente, viii
- array, 82, 92, 152–154, 166, 171, 230, 234, 236, 251–253, 371, 372, 389, 390, 396, 397, 399, 414, 529
- array, 87, 94, 155, 230–232, 237, 238, 240, 251, 262, 263, 267–269, 290
- article, viii, 13, 33, 37, 48, 51, 52, 56, 58, 60, 62–65, 68, 100, 169, 184, 375, 430, 577
- article, book, letter, report, slides, 169
- AU<sub>T</sub>E<sub>X</sub>, 500–502, 509–514
- Aufzählungen, 71
- Auslassungspunkte, 554
- aux-Datei, 109, 174, 267, 304, 313
- avant, 348, 350
- babel, 210
- babel, ix, 55, 58, 205, 206, 208, 210, 214, 216, 218, 219, 226, 301, 375, 425, 447, 460–462, 470, 474, 552
- Basispakete, 169
- bbl-Datei, 448, 449, 452, 464
- beamer, 48
- Befehle, viii
- Beispiele, viii, ix
- Benutzeroberflächen, 499
- Bezugsquellen
  - Software, 10
- bfquote, 24
- bib-Datei, 448, 452, 530
- bibchapter, 450
- bibdiv, 450
- Bibliografie
  - Felder
    - address, 460, 463, 466–469
    - author, 459, 466, 468–470, 482
    - booktitle, 465, 468
    - book, 465, 467, 468, 478
    - collection, 479
    - conference, 461, 465–468, 478
    - contribution, 464, 466–469, 478, 480
    - date, 453, 460, 462, 465–468
    - doi, 468, 479
    - edition, 466–469, 479, 481
    - editor, 459, 466–470, 482
    - eprint, 462, 468, 469, 483
    - hyphenation, 460, 461, 468
    - institution, 460, 469
    - isbn, 468, 480
    - issn, 468
    - journal, 467, 468, 483
    - label, 461, 468
    - language, 460, 461, 468
    - month, 468
    - note, 466–468
    - number, 463, 467–469
    - organization, 460, 462–464, 466–469
    - pages, 467, 468, 483
    - partial, 467, 468, 480
    - part, 466–469
    - place, 460, 469
    - proceedings, 479
    - publisher, 466–468
    - reprint, 468, 480
    - review, 468, 480
    - school, 460, 469
    - series, 466–469, 481
    - setup, 461, 468
    - status, 468
    - subtitle, 466–468
    - title, 461, 466–468
    - translation, 468, 480
    - translator, 466–470
    - type, 466, 467, 469, 480

- volume, 462, 465–468, 479, 481
- xref, 455, 461, 468
- year, 460, 468
- Stile**
  - abbrv, 444
  - alpha, 443, 444
  - plain, 444
  - unsrt, 444
- Typen**
  - Neuer Typ, 477
  - article, 459, 462, 467, 468, 479
  - book, 459, 468, 469, 477, 481
  - collection, 465, 476
  - innerbook, 478
  - manual, 477
  - miscellaneous, 459, 477
  - periodical, 476
  - proceedings, 477
  - report, 459, 468, 469, 477
  - techreport, 477
  - thesis, 459, 468, 469, 480
  - unpublished, 477
- biblist, 450, 451
- bibsection, 450
- BIB $\TeX$** , 428, 442–446, 448, 449, 451–459, 461–465, 468, 469, 472, 500, 508, 509, 514, 524, 530, 532
- Bindestrich**, 122
- binom, 402
- bm, 240, 243–245, 357, 368
- Bmatrix, 372, 396
- bmatrix, 372, 396
- bmp-Datei, 321
- book, 13, 33, 50–52, 58, 60, 62, 67, 68, 100, 101, 375, 430, 526, 577
- bookman, 348, 351
- Boxen**, 124
  - ausgefüllte, 130
  - horizontal, 124
  - Parameter, 127
  - speichern, 127
  - verschieben, 128
  - vertikal, 129
- bp Maßeinheit, 131
- calc, 161, 246, 247, 300
- caption, 271
- cases, 372, 389, 391
- cc Maßeinheit, 131
- CD, 368, 420, 421
- center, 56, 95, 129
- cfrac, 402
- chancery, 348, 351
- charter, 348, 352
- chktex, 529
- Classes, *siehe* Klassen
- clp-Datei, 321
- cm Maßeinheit, 131
- cmmib57, 494
- color, 55, 287, 317, 318, 320, 323, 324, 328, 344, 346
- colordvi, 322
- comment, 311
- Comprehensive  $\TeX$  Archive Network**, 11
- Counter**, *siehe* Zähler
- courier, 348, 352
- CTAN, 11
- CUA, 506
- DANTE e. V.**, 11
- Darstellung**
  - Argumente, viii
  - Befehle, viii
  - Beispiele, viii
  - Eingabe, viii
  - Klassen, viii
  - Optionen, viii
  - Pakete, viii
  - Programmbeispiele, viii
  - Umgebungen, viii
  - Zeichensatz, ix
- Datei**
  - aux, 109, 174, 267, 304, 313
  - bbl, 448, 449, 452, 464
  - bib, 448, 452, 530
  - bmp, 321
  - clp, 321
  - dtx, 54, 172
  - dvi, 196, 529, 532
  - emf, 321
  - eps, 344, 347
  - eps.gz, 344
  - gif, 321, 540

- idx, 190
- ind, 190
- ins, 54
- jpeg, 321, 540
- jpg, 321
- lof, 107, 108, 174
- log, 16, 200, 201, 265, 269, 292, 300, 504, 532
- lot, 107, 108, 174
- ltb, 448, 449, 452–454, 470
- mps, 321
- mSP, 321
- pcl, 321
- pcx, 321
- pdf, 321, 540
- pict, 321
- picture, 321
- pk, 175
- png, 321, 540
- pnm, 540
- pntg, 321
- ppm, 540
- ps, 18, 320, 321, 344, 347
- ps.gz, 344
- rps, 321
- sty, 54, 64
- svg, 540
- targa, 321
- tex, 581
- tga, 321
- thm, 434
- tif, 321
- tiff, 321, 540
- tm, 533
- toc, 38, 68, 69, 174
- wmf, 321
- xpm, 540
- Dateinamenserweiterung, 54, 581
- dbinom, 402
- dcolumn, 248, 249, 267, 268
- delarray, 251–253
- description, 39, 41, 451
- dfrac, 402
- Dialekt, 212
- dinglist, 360
- displaymath, 140, 141, 371, 382, 384
- doc, 172, 194
- document, 28, 448
- Dokumentklassen, 33
- Dokumentlayout, 47
- Dokumentstrukturen, 47
- Driver, *siehe* Treiber
- dtX-Datei, 54, 172
- dvi-Datei, 196, 529, 532
- dvi2ps, 319
- dvialw, 319
- dvilaser/ps, 319
- dvipdf, 319
- dvipdfm, 319
- dvips, ix
- dvips, 319, 514
- dvipsone, 319
- dvitops, 319
- dviwin, 319
- dviwindo, 319
- E-Mail-Adressen, 554
- Editor, 499
- Eingabe, viii
- Einheiten, 547
- $\epsilon$ m Maßeinheit, 131
- em, 119
- em $\TeX$ , 319
- Emacs, 501–511
- Emacs, 501
- emf-Datei, 321
- Entwicklungsumgebung, 514
- enumerate, 39, 40, 71, 253, 360
- enumerate, 253, 254
- eps-Datei, 344, 347
- eps.gz-Datei, 344
- epsfig, 317, 318, 344–346
- eqnarray, 141–143, 145, 171, 371, 385, 386
- eqnarray\*, 142, 143, 145, 371, 386
- equation, 141, 371, 376, 378, 382, 384, 390
- equation\*, 371
- Eszett, 30, 156
- $\epsilon$ - $\TeX$ , 7, 300
- eucal, 429, 483, 492
- eufrak, 483, 490
- euler, 493
- euscript, 429, 483

- ex Maßeinheit, 131
- expdlist, 254
- exscale, 173, 176, 355, 357
- ε<sub>X</sub>TeX, 7
- fancybox, 128
- fancyhdr, 53
- figure, 103, 107, 274, 328, 540
- figure\*, 103, 276
- fix-cm, 173, 175
- fixltx2e, 173, 174
- flafter, 106, 176
- flalign, 371
- flalign\*, 371, 389
- float, 103, 104, 106
- flushleft, 95
- flushright, 95
- foiltex, 322
- fontenc, 55, 114, 115, 177, 178, 197
- fontsmpl, 255
- Formelsatz, 369
- Fortsetzungspunkte, 554
- fourier, 348, 366
- frac, 402
- fred, 289
- FreePDF, 6
- ftnright, 256–258, 273
- Fußnote, 98
- Fußzeilen, 52
- gather, 371, 372, 376, 378, 385, 386, 388, 392
- gather\*, 371, 385
- gathered, 371, 384, 389, 392, 393
- Gedankenstrich, 122
- genfrac, 402
- german, viii
- german, 31, 32, 35, 474, 599
- german.sty, 599
- geschachtelte Klammern, 565
- GhostScript, 347
- Ghostview, 6, 516, 538
- gif-Datei, 321, 540
- Gleitobjekte, 102
- Gliederungsbefehle, 59
- GNU Emacs, *siehe* Emacs
- GNUplot, 542, 543
- gnuplot, 537, 541
- Grafische Oberflächen, *siehe* Oberflächen
- graphics, 317, 318, 328, 334–336, 345, 346
- graphics:, 330, 334
- graphicx, ix, 54, 317, 318, 328, 334–336, 345, 346
- graphicx:, 330, 334
- graphpap, 179
- graphpap, 55, 178
- Gravis, 32
- Gruppenklammer, 138
- GUILE, 540
- Gutenberg, 10
- halbe Anführungszeichen, 551
- harvard, 287
- helvet, 348, 353
- Hervorhebung, 119
- hhline, 94, 258, 261–263, 267, 269
- Himmelsrichtungen, 547
- Hochkomma, *siehe* Anführungszeichen
- HTML, v, 7, 501, 523, 529, 532, 533, 538
- hyperref, 443, 444
- ispell, 205
- IDE, 514
- idx-Datei, 190
- ifoption, 428
- ifthen, 55, 180, 181, 184
- in Maßeinheit, 131
- ind-Datei, 190
- indentfirst, 263
- index, 441
- Inhaltsverzeichnis, 68
- inputenc, 31, 55, 186, 187, 425, 447
- ins-Datei, 54
- ispell, 514, 529
- itemize, 39, 71–73, 218, 221, 222, 360, 363, 531
- jpeg-Datei, 321, 540
- jpg-Datei, 321

- KT<sub>E</sub>Xmaker2, 521
- KDE, 18
- keyval, 317, 318, 344, 346, 369, 428
- KGhostview, 516
- Kile, v, 515, 521, 522
- Klammern, 565
- Klassen, viii
  - amsart, 48
  - amsart, 376, 430, 431, 433, 440
  - amsbook, 49
  - amsbook, 376, 430, 431, 433, 440
  - amstxt, 49
  - amstxt, 430
  - amslatex, 49
  - amslatex, 430
  - amstxt, 49
  - amstxt, 376, 430
  - amstxt, 433
  - article, 48
  - article, viii, 13, 33, 37, 48, 51, 52, 56, 58, 60, 62–65, 68, 100, 169, 184, 375, 430, 577
  - article, book, letter, report, slides, 169
  - beamer, 48
  - book, 48
  - book, 13, 33, 50–52, 58, 60, 62, 67, 68, 100, 101, 375, 430, 526, 577
  - german, viii
  - letter, 48
  - letter, 33, 60, 184
  - ltxdoc, 48
  - ltxdoc, 172
  - ltxguide, 48
  - ltxnews, 48
  - minimal, 48
  - proc, 48
  - proc, 60, 169
  - report, 48
  - report, 13, 33, 48, 51, 52, 56, 58, 60, 62, 68, 100, 375, 430, 577
  - scrartcl, 49
  - scrbook, 49
  - scrlettr, 49
  - scrreprt, 49
  - slides, 48
  - slides, 33, 60
- Kodierung, 111, 114, 177, 186
- KOMA-Script, 34, 49, 214, 226, 545
- Kopfzeilen, 52
- KPDF, 516
- kuvio, 420
- lacheck, 514
- landscape, 318, 345
- L<sub>A</sub>T<sub>E</sub>X-Eingabe, viii
- L<sub>A</sub>T<sub>E</sub>X-Zeichensatz, 112
- L<sub>A</sub>T<sub>E</sub>X, 7
- L<sub>A</sub>T<sub>E</sub>X3, 8
- L<sub>A</sub>T<sub>E</sub>X 2<sub>ε</sub>, 8
- latexsym, 55, 155, 157–159, 166, 167, 175, 189, 194
- Layout, 33, 47
- layout, 265
- letter, 33, 60, 184
- Ligaturen, 557
- Linux, 18
- list, 71, 74, 77
- Listen, 71, 74
- Listings, 170
- Literatur, 591
- ln03, 319
- lof-Datei, 107, 108, 174
- log-Datei, 16, 200, 201, 265, 269, 292, 300, 504, 532
- longtable, 78, 82, 266–273
- longtable, 266–269, 272
- lot-Datei, 107, 108, 174
- lscape, 318, 344, 345
- ltb-Datei, 448, 449, 452–454, 470
- ltxdoc, 172
- L<sub>Y</sub>X, 523
- L<sub>Y</sub>X, 9, 500, 523–528, 530–533, 630, 631
- Längenangaben, 133
- MakeIndex, 190, 441, 514
- makeidx, 55, 190, 432, 441
- Makro, 22
- Makropakete, 54
- Mathematische Symbole, 166

Mathematischer Satz, 367, 369,  
     569  
 mathpazo, 348, 354, 355, 365, 366,  
     484  
 mathpple, 365  
 mathptm, 365, 366  
 mathptmx, 348, 356, 357, 365,  
     366  
 mathscinet, 428, 445  
 matrix, 372, 396, 397  
 Maßangaben, 130  
 Maßeinheiten, 130  
 minipage, 94, 98, 129, 221, 246, 272,  
     275, 288, 531, 539  
 mm Maßeinheit, 131  
 mps-Datei, 321  
 msp-Datei, 321  
 mu, 401  
 multicol, 256–258, 273, 274, 276,  
     282  
 multicol, 185, 274–276, 282–  
     285  
 multicol, 278, 281  
 multicol\*, 275, 276  
 multind, 440  
 multiols, 279  
 multiline, 371, 378, 382–384,  
     390  
 multiline\*, 371, 382  
  
 natbib, 287  
 newcent, 348, 358  
 newlfont, 193, 194  
 NFSS, 193, 194, 200  
 ngerman, 31, 32, 35, 55, 58, 107, 375,  
     474, 599  
 ngerman.sty, 599  
 NTG, 10  
 Nummern, 562  
  
 Oberfläche  
     L<sub>A</sub>T<sub>E</sub>X, 523  
     T<sub>E</sub>Xmacs, 533  
     Emacs, 501  
     Emacs mit AUCT<sub>E</sub>X, 509  
     grafische, 514  
     textorientiert, 500  
     WYSIWYG, 522  
 oldlfont, 193

Optionen, viii  
     10pt, 33, 50, 132, 431, 432  
     11pt, 33, 50, 132, 431  
     12pt, 33, 50, 132, 431  
     8pt, 431  
     9pt, 431  
     a4paper, 33, 50, 431  
     a5paper, 50  
     acadian, 207, 219  
     afrikaans, 207, 301  
     alphabetic, 443, 450, 463  
     american, 207, 216, 301  
     angle, 339  
     article, 458  
     australian, 207, 216  
     austrian, 207, 210, 213,  
         301  
     author-year, 443, 450, 473,  
         474  
     b5paper, 50  
     bahasa, 207  
     balancingshow, 274, 285  
     bar, 289  
     basque, 207  
     bb, 336  
     brazil, 208, 301  
     brazilian, 208, 265  
     breton, 207, 301  
     british, 207, 216  
     bulgarian, 207  
     canadian, 207, 216  
     canadien, 219  
     catalan, 207, 301  
     centertags, 431, 432  
     change, 296  
     citation-order, 445  
     compressed-cites, 443  
     croatian, 207, 301  
     czech, 208, 301  
     danish, 207, 301  
     debugshow, 201, 276, 285  
     draft, 33, 50, 302, 309, 329, 338,  
         431  
     dutch, 207, 265, 301  
     dvipdfm, 54  
     dvips, 323  
     e-only, 431  
     english, 207, 216, 265, 301  
     errorshow, 201, 285

- esperanto, 207, 301
- estonian, 207
- executivepaper, 50
- final, 50, 287, 302, 309, 431, 432
- finnish, 207, 301
- fleqn, 51, 140, 141, 144, 145, 287, 382, 431
- fleqn, leqno, proc, tlenc, 169
- francais, 207, 218, 219, 265, 301
- french, 207, 219, 265, 301, 552
- frenchb, 207, 218, 219
- full, 197
- galician, 207, 301
- german, 207, 210, 213, 265, 301
- germanb, 207, 210, 213, 301
- greek, 207, 301
- headings, 53, 60
- hebrew, 207
- height, 337–339
- hiresbb, 329, 336
- hungarian, 208
- icelandic, 207
- infoshow, 201, 274, 285
- interlingua, 207
- intllimits, 370
- irish, 207
- italian, 207, 265, 301
- landscape, 431
- latin, 207
- latin1, 189
- legalpaper, 50
- leqno, 51, 382, 431, 432
- letterpaper, 50, 431, 432
- loading, 201
- logical-quotes, 445, 478
- lowersorbian, 207
- magyar, 208, 301
- makeidx, 431
- margin, 296
- markshow, 274, 285
- monochrome, 328
- naustrian, 207, 210, 213, 301
- newzealand, 207, 216
- ngerman, 58, 207, 210, 213, 265, 301
- noamsfonts, 431
- noBBpl, 354
- nobysame, 445
- nomath, 431
- non-compressed-cites, 443
- non-sorted-cites, 443
- norsk, 207, 301
- nosumlimits, 370
- notitlepage, 51, 56, 58, 431
- numeric, 442, 443, 451
- nynorsk, 207, 301
- OML, 178
- OMS, 178
- onecolumn, 431, 432
- oneside, 50, 431
- only, 286, 289
- openany, 431
- openright, 50, 431
- origin, 331
- OT1, 178
- OT4, 178
- plain, 53, 296
- polish, 208, 301
- polutonikogreek, 207
- portrait, 431, 432
- portuges, 208, 301
- portuguese, 208, 265
- proc, 169
- psamsfonts, 431, 484, 487, 490, 495
- reqno, 382, 431
- romanian, 208, 301
- russian, 208, 301
- safe, 197
- samin, 207
- scottish, 208
- serbian, 208
- shortalphabetic, 443, 450, 463
- slantedGreek, 355, 357
- slovak, 208, 301
- slovene, 208, 301
- sorted, 445
- sorted-cites, 443
- spanish, 208, 265, 301

- sumlimits, 370
- swedish, 208, 301
- Tl, 178
- tbtags, 431
- Tex Auto Save, 511
- Tex Parse Self, 511
- titlepage, 51, 56, 58, 431
- totalheight, 338, 339
- traditional-quotes, 445
- TS1, 178
- turkish, 208, 301
- twocolumn, 174, 185, 256, 273, 431
- twoside, 50, 101, 431, 432, 577
- type, 338, 339
- UKenglish, 207, 216
- ukrainian, 208
- units=-360, 330
- units=6.283185, 331
- uppertorbian, 207
- usenames, 323
- USenglish, 207, 216
- utf8, 189
- warningshow, 201
- welsh, 208
- width, 339
- x, 331
- y, 331
- y2k, 443, 450
- otherlanguage, 209
- OzTeX**, 319
- Packages**, *siehe* Pakete
- Pakete**, viii
  - accents, 417
  - afterpage, 229
  - alltt, 55, 170
  - amsbsy, 243, 368, 369, 421, 430, 485, 487
  - amscd, 368, 420
  - amsmath, 163, 167, 176, 189, 429, 430, 432, 483, 484, 487, 490, 492, 495
  - amsgen, 421, 430
  - amsjpa, 446
  - amsmath, 138, 150, 151, 158, 166, 176, 369, 372–377, 380, 381, 389, 391, 396, 399, 401–404, 406–408, 410–414, 416–421, 423–425, 428, 430, 431, 434, 435, 485, 530
  - amsmidx, 430, 440, 441
  - amsopn, 157, 368, 369, 423, 430
  - amsrefs, 428, 430, 442, 444–453, 455–459, 461, 463–465, 467–469, 472–474, 476, 477
  - amsrefs, 459
  - amssymb, 155, 157, 158, 166, 189, 429, 483–485, 487–490
  - amstex, 5, 55, 374, 376, 426, 485
  - amstext, 368, 369, 425, 430, 570
  - amsthm, 63, 294, 430, 433, 435
  - amsxtra, 426
  - array, 87, 94, 155, 230–232, 237, 238, 240, 251, 262, 263, 267–269, 290
  - avant, 348, 350
  - babel, ix, 55, 58, 205, 206, 208, 210, 214, 216, 218, 219, 226, 301, 375, 425, 447, 460–462, 470, 474, 552
  - bm, 240, 243–245, 357, 368
  - bookman, 348, 351
  - calc, 161, 246, 247, 300
  - caption, 271
  - chancery, 348, 351
  - charter, 348, 352
  - cmmib57, 494
  - color, 55, 287, 317, 318, 320, 323, 324, 328, 344, 346
  - colordvi, 322
  - courier, 348, 352
  - CUA, 506
  - dcolum, 248, 249, 267, 268
  - delarray, 251–253
  - doc, 172, 194
  - dvips, ix
  - enumerate, 253, 254
  - epsfig, 317, 318, 344–346
  - eucal, 429, 483, 492
  - euftrak, 483, 490
  - euler, 493
  - euscript, 429, 483

- expdlist, 254
- exscale, 173, 176, 355, 357
- fancybox, 128
- fancyhdr, 53
- fix-cm, 173, 175
- fixltx2e, 173, 174
- flafter, 106, 176
- float, 103, 104, 106
- foiltex, 322
- fontenc, 55, 114, 115, 177, 178, 197
- fontsmpl, 255
- fourier, 348, 366
- fred, 289
- ftnright, 256–258, 273
- german, 31, 32, 35, 474, 599
- german.sty, 599
- graphics, 317, 318, 328, 334–336, 345, 346
- graphics:, 330, 334
- graphicx, ix, 54, 317, 318, 328, 334–336, 345, 346
- graphicx:, 330, 334
- graphpap, 55, 178
- harvard, 287
- helvet, 348, 353
- hhline, 94, 258, 261–263, 267, 269
- hyperref, 443, 444
- ifoption, 428
- ifthen, 55, 180, 181, 184
- indentfirst, 263
- index, 441
- inputenc, 31, 55, 186, 187, 425, 447
- keyval, 317, 318, 344, 346, 369, 428
- KOMA-Script, 34, 49, 214, 226, 545
- kuvio, 420
- latexsym, 55, 155, 157–159, 166, 167, 175, 189, 194
- layout, 265
- longtable, 266–269, 272
- lscope, 318, 344, 345
- makeidx, 55, 190, 432, 441
- mathpazo, 348, 354, 355, 365, 366, 484
- mathpple, 365
- mathptm, 365, 366
- mathptmx, 348, 356, 357, 365, 366
- mathscinet, 428, 445
- multicol, 256–258, 273, 274, 276, 282
- multicols, 278, 281
- multind, 440
- natbib, 287
- newcent, 348, 358
- newlfont, 193, 194
- ngerman, 31, 32, 35, 55, 58, 107, 375, 474, 599
- ngerman.sty, 599
- oldlfont, 193
- palatino, 365, 366
- parskip, 34
- path, 557
- pcatcode, 428
- pifont, 358
- plain, 5, 35, 139, 140, 147, 148, 158, 407
- psfig, 317, 344–346
- psnfss, 347–349
- pstcol, 318, 344, 346
- pstricks, 318, 344, 346
- rawfonts, 194, 286, 288
- rkeyval, 428
- scraddr, 49
- scrdate, 49
- scrfile, 49
- scrpage2, 49
- scrtime, 49
- shortvrb, 194
- showidx, 55, 190–192
- showkeys, 286, 313
- somedefs, 286, 288, 289
- somefonts, 288
- syntonly, 196
- tabularx, 290, 292, 293
- textcmds, 428, 430, 439, 445
- textcomp, 55, 73, 128, 196, 197, 556, 565
- thb, 296
- thc, 295
- thcb, 296
- theorem, 63, 294, 433
- thm, 295
- thmb, 296

- thp, 295
- times, 256, 365, 366
- trace, 299, 300
- tracefont, 200, 201
- trig, 317, 318, 344, 346
- typearea, 49
- uescript, 492
- utopia, 365, 366
- varioref, 181, 287, 301, 304, 306, 308
- verbatim, 309
- xr, 313
- xspace, 314, 315
- xypic, 420
- palatino, 365, 366
- parskip, 34
- path, 557
- pc Maßeinheit, 131
- pcatcode, 428
- pcl-Datei, 321
- pcx-Datei, 321
- PDF, v, 18, 19, 322, 328, 529, 532, 533, 538
- pdf-Datei, 321, 540
- pdfTeX, 319
- pict-Datei, 321
- picture, 178
- picture, 55, 178, 179
- picture-Datei, 321
- pifont, 358
- Pilist, 363
- pk-Datei, 175
- plain, 5, 35, 139, 140, 147, 148, 158, 407
- pmatrix, 372, 396
- png-Datei, 321, 540
- pnm-Datei, 540
- pntg-Datei, 321
- PostScript, vii, ix, 6, 18, 131, 197, 214, 226, 241, 317–319, 321, 322, 328, 334, 335, 339, 344–349, 354, 358, 362, 363, 365, 366, 429, 432, 447, 484, 487, 490, 492, 494, 495, 514, 516, 527, 529, 532, 533, 538, 540, 555, 556, 599
- ppm-Datei, 540
- proc, 60, 169
- Programmbeispiele, viii
- Programme
- Acrobat Reader, 18
- AUcTeX, 500–502, 509–514
- BIBTeX, 428, 442–446, 448, 449, 451–459, 461–465, 468, 469, 472, 500, 508, 509, 514, 524, 530, 532
- chktex, 529
- dvi2ps, 319
- dvi2l, 319
- dvi2laser/ps, 319
- dvi2pdf, 319
- dvi2pdfm, 319
- dvips, 319, 514
- dvipsone, 319
- dvi2tops, 319
- dviwin, 319
- dviwindo, 319
- emTeX, 319
- Emacs, 501–511
- $\epsilon$ -TeX, 7, 300
- $\epsilon_X$ TeX, 7
- FreePDF, 6
- GhostScript, 347
- Ghostview, 6, 516, 538
- GNUplot, 542, 543
- gnuplot, 537, 541
- GUILE, 540
- HTML, v, 7, 501, 523, 529, 532, 533, 538
- ispell, 205
- ispell, 514, 529
- KTeXmaker2, 521
- KDE, 18
- KGhostview, 516
- Kile, v, 515, 521, 522
- KPDF, 516
- lacheck, 514
- Linux, 18
- ln03, 319
- L<sub>A</sub>TeX, 9, 500, 523–528, 530–533, 630, 631
- MakeIndex, 190, 441, 514
- OzTeX, 319
- PDF, v, 18, 19, 322, 328, 529, 532, 533, 538
- pdfTeX, 319
- PostScript, vii, ix, 6, 18, 131, 197, 214, 226, 241, 317–319, 321, 322, 328, 334, 335, 339, 344–349, 354,

- 358, 362, 363, 365, 366, 429, 432, 447, 484, 487, 490, 492, 494, 495, 514, 516, 527, 529, 532, 533, 538, 540, 555, 556, 599
- psprint, 319
- PTI Laser/HP, 319
- PTI Laser/PS, 319
- PUBps, 319
- SCHEME, 533, 536, 540
- T<sub>E</sub>Xmacs, 500, 523, 533–538, 540, 542, 543, 631
- T<sub>E</sub>Xmaker, 500, 514, 515, 519, 521, 522
- T<sub>E</sub>XnicCenter, v, 9, 499, 500, 515, 519–522
- Textures, 319
- truetex, 319
- Unix, 65
- V<sub>T</sub>E<sub>X</sub>, 319
- vi, 506
- Windows, 18, 65
- WinEdt, v, 499
- xdvi, 17, 319, 513
- XEmacs, 502
- Xfig, 540
- XForms, 524, 525
- XML, 7, 501, 523, 533
- Xtem, 500
- X<sub>T</sub>E<sub>X</sub>Shell, 500
- YAP, 18
- proof, 434–436
- Präambel, 34, 104
- ps-Datei, 18, 320, 321, 344, 347
- ps.gz-Datei, 344
- psfig, 317, 344–346
- psnfss, 347–349
- psprint, 319
- pstcol, 318, 344, 346
- pstricks, 318, 344, 346
- pt Maßeinheit, 131
- PTI Laser/HP, 319
- PTI Laser/PS, 319
- PUBps, 319
- Querverweise, 108
- quotation, 96
- quote, 23–25, 44, 96
- raggedleft, 129
- Randnote, 98
- Randnotiz, 101
- rawfonts, 194, 286, 288
- Rechteckboxen, 130
- Rechtschreibprüfung, 205
- Rechtschreibung, 210
- Referenzen, 108
- report, 13, 33, 48, 51, 52, 56, 58, 60, 62, 68, 100, 375, 430, 577
- rkeyval, 428
- rps-Datei, 321
- Satzzeichen, 554
- scharfes s, *siehe* Eszett
- SCHEME, 533, 536, 540
- Schreibmaschinenschrift, 170
- Schriftbefehle, 118
  - Kombinationen, 118
- Schriften, 111
- Schriftfamilie, 113
- Schriftform, 113
- Schriftgröße, 114
  - wechseln, 117
- Schriftstärke, 114
- Schriftumgebungen, 117
- scraddr, 49
- scrdate, 49
- scrfile, 49
- scrpage2, 49
- scrtime, 49
- Seitennummerierung, 53
- shortvrb, 194
- showidx, 55, 190–192
- showkeys, 286, 313
- slides, 33, 60
- smallmatrix, 372, 396, 397, 399
- Software, 9
  - Bezugsquellen, 10
  - Verteilung, 11
- somedefs, 286, 288, 289
- somefonts, 288
- Sonderzeichen, 120, 121, 554
- sp Maßeinheit, 131
- Spalten, 230
- Spationierung, 567
- split, 370, 371, 382, 389–391, 393, 411
- Sprachunterstützung, 205

- Striche, 563
- sty-Datei, 54, 64
- subarray, 414, 415
- subequations, 377, 379
- supertabular, 78, 82
- svg-Datei, 540
- Symbole
  - mathematische, 166
- syntonly, 196
- tabbing, 41, 78, 80, 82, 103, 269
- Tabellen, 230
- table, 77, 103, 107, 267, 268, 270, 274, 328, 539
- table\*, 103, 276
- tabular, viii, 44, 57, 78, 82, 83, 85, 91, 103, 141, 152–154, 171, 230, 234, 236, 239, 246, 248, 258, 267–272, 290, 291, 293, 399, 539, 542
- tabular\*, 85, 87, 90, 152, 268, 272, 290, 291
- tabularx, 290–293
- tabularx, 290, 292, 293
- targa-Datei, 321
- tbinom, 402
- Teilformel, 138, 389
- $\TeX$ -Zeichensatz, 111
- $\TeX$ , 4
- tex-Datei, 581
- $\TeX$ macs, 500, 523, 533–538, 540, 542, 543, 631
- $\TeX$ maker, 500, 514, 515, 519, 521, 522
- $\TeX$ nicCenter, v, 9, 499, 500, 515, 519–522
- Text
  - hervorheben, 119
- textcmds, 428, 430, 439, 445
- textcomp, 55, 73, 128, 196, 197, 556, 565
- Texteditor, 499
- Textstrukturen, 71, 111
- Textures, 319
- tfrac, 402
- tga-Datei, 321
- thb, 296
- thc, 295
- thcb, 296
- thebibliography, 442, 449, 450
- theorem, 294, 295
- theorem, 63, 294, 433
- thm-Datei, 434
- thm, 295
- thmb, 296
- thp, 295
- tif-Datei, 321
- tiff-Datei, 321, 540
- times, 256, 365, 366
- Titelseite, 56
- titlepage, 52, 56, 58
- tm-Datei, 533
- toc-Datei, 38, 68, 69, 174
- Tools, 229
- trace, 299, 300
- tracefmt, 200, 201
- Treiber
  - dvi2ps, 319, 320
  - dviawl, 319, 320
  - dvilaser, 319, 320
  - dvipdf, 319, 321
  - dvipdfm, 319, 321
  - dvips, 319, 321, 323
  - dvipsone, 319, 321
  - dvitops, 319, 320
  - dviwin, 319, 321
  - dviwindo, 319, 321
  - emtex, 319, 321
  - ln, 319, 320
  - oztex, 319, 321
  - pctex32, 319, 321
  - pctexhp, 319, 321
  - pctexps, 319, 321
  - pctexwin, 319, 321
  - pdftex, 319, 321
  - psprint, 319, 320
  - pubps, 319, 320
  - tcidvi, 319, 321
  - textures, 319, 321
  - truettex, 319, 321
  - vtex, 319, 321
  - xdvi, 319, 321
- Trennregeln, 210
- Trennung, 123
- trig, 317, 318, 344, 346
- trivlist, 71, 77, 129

- truertex, 319
- TUG, 10
- typearea, 49
- Typografische Konventionen, vii
- uescript, 492
- Umgebungen, viii
  - abel, 450
  - abstract, 55, 56, 58
  - align, 371, 372, 386–390, 393, 394
  - align\*, 371
  - alignat, 371, 372, 387, 388, 394
  - alignat\*, 371, 388
  - aligned, 371, 389, 393–395
  - alignedat, 371, 389, 394, 395
  - alltt, 170, 171
  - amsrefs, 451
  - array, 82, 92, 152–154, 166, 171, 230, 234, 236, 251–253, 371, 372, 389, 390, 396, 397, 399, 414, 529
  - babel, 210
  - bfquote, 24
  - bibchapter, 450
  - bibdiv, 450
  - biblist, 450, 451
  - bibsection, 450
  - binom, 402
  - Bmatrix, 372, 396
  - bmatrix, 372, 396
  - cases, 372, 389, 391
  - CD, 368, 420, 421
  - center, 56, 95, 129
  - cfrac, 402
  - comment, 311
  - dbinom, 402
  - description, 39, 41, 451
  - dfrac, 402
  - dinglist, 360
  - displaymath, 140, 141, 371, 382, 384
  - document, 28, 448
  - em, 119
  - enumerate, 39, 40, 71, 253, 360
  - eqnarray, 141–143, 145, 171, 371, 385, 386
  - eqnarray\*, 142, 143, 145, 371, 386
  - equation, 141, 371, 376, 378, 382, 384, 390
  - equation\*, 371
  - figure, 103, 107, 274, 328, 540
  - figure\*, 103, 276
  - flalign, 371
  - flalign\*, 371, 389
  - flushleft, 95
  - flushright, 95
  - frac, 402
  - gather, 371, 372, 376, 378, 385, 386, 388, 392
  - gather\*, 371, 385
  - gathered, 371, 384, 389, 392, 393
  - genfrac, 402
  - graphpap, 179
  - itemize, 39, 71–73, 218, 221, 222, 360, 363, 531
  - landscape, 318, 345
  - list, 71, 74, 77
  - longtable, 78, 82, 266–273
  - matrix, 372, 396, 397
  - minipage, 94, 98, 129, 221, 246, 272, 275, 288, 531, 539
  - multicols, 185, 274–276, 282–285
  - multicols\*, 275, 276
  - multiols, 279
  - multiline, 371, 378, 382–384, 390
  - multiline\*, 371, 382
  - otherlanguage, 209
  - picture, 55, 178, 179
  - Pilist, 363
  - pmatrix, 372, 396
  - proof, 434–436
  - quotation, 96
  - quote, 23–25, 44, 96
  - raggedleft, 129
  - smallmatrix, 372, 396, 397, 399
  - split, 370, 371, 382, 389–391, 393, 411

- subarray, 414, 415
- subequations, 377, 379
- supertabular, 78, 82
- tabbing, 41, 78, 80, 82, 103, 269
- table, 77, 103, 107, 267, 268, 270, 274, 328, 539
- table\*, 103, 276
- tabular, viii, 44, 57, 78, 82, 83, 85, 91, 103, 141, 152–154, 171, 230, 234, 236, 239, 246, 248, 258, 267–272, 290, 291, 293, 399, 539, 542
- tabular\*, 85, 87, 90, 152, 268, 272, 290, 291
- tabularx, 290–293
- tbinom, 402
- tfrac, 402
- thebibliography, 442, 449, 450
- theorem, 294, 295
- titlepage, 52, 56, 58
- trivlist, 71, 77, 129
- verbatim, 55, 170, 171, 195, 309–312
- verbatim\*, 309–311
- verse, 97
- Vmatrix, 372, 396
- vmatrix, 372, 396–398
- xxquote, 25, 26
- Umlaute, 30, 156
- Uniform Resource Locator, 465
- Unix, 65
- Unterstreichen, 120
- utopia, 365, 366
- V<sub>T</sub><sub>E</sub>X, 319
- varioref, 181, 287, 301, 304, 306, 308
- verbatim, 55, 170, 171, 195, 309–312
- verbatim, 309
- verbatim\*, 309–311
- Vermaßung, 133
- verse, 97
- vi, 506
- Vmatrix, 372, 396
- vmatrix, 372, 396–398
- Windows, 18, 65
- WinEdt, v, 499
- wmf-Datei, 321
- Worttrennung, 205
- Worttrennungen, 572
- Wortzwischenräume, 567
- xdvi, 17, 319, 513
- XEmacs, 502
- Xfig, 540
- XForms, 524, 525
- XML, 7, 501, 523, 533
- xpm-Datei, 540
- xr, 313
- xspace, 314, 315
- Xtem, 500
- X<sub>T</sub><sub>E</sub>XShell, 500
- xxquote, 25, 26
- xypic, 420
- YAP, 18
- Zahlen, 559
- Zeichensatz, 111
  - L<sub>A</sub>T<sub>E</sub>X, 112
  - T<sub>E</sub>X, 111
- Zeichensatzbefehle, 115
- Zeichensätze
  - verwendete, ix
- Zeilenumbruch, 574
- Ziffern, 559
- Zitate, 550
- Zusammenfassungen, 55
- Zähler, 134
  - bib, 451
  - collectmore, 284
  - columnbadness, 285
  - equation, 378
  - finalcolumnbadness, 285
  - MaxMatrixCols, 396, 397
  - page, 181
  - parentequation, 378
  - part, 220
  - section, 181
  - subsection, 375
  - tracingmulticols, 275, 285
  - unbalance, 285