The Not So Short Introduction to $\LaTeX\,2_\varepsilon$

Or LaTeX2e in 62 minutes

by Tobias Oetiker
Hubert Partl, Irene Hyna and Elisabeth Schlegl

Version 2.1, 16th July, 1995

Thank you!

Much of the material used in this introduction comes from an Austrian introduction to LaTeX 2.09 written in German by:

> Hubert Partl <partl@mail.boku.ac.at>
> *Zentraler Informatikdienst der Universität für Bodenkultur Wien*
>
> Irene Hyna <Irene.Hyna@bmwf.ac.at>
> *Bundesministerium für Wissenschaft und Forschung Wien*
>
> Elisabeth Schlegl <no email>
> *in Graz*

If you are interested in the German document you can find a version updated for LaTeX 2_ε by Jörg Knappen at `CTAN:/tex-archive/info/lkurz`

While preparing this document I asked for reviewers on `comp.text.tex`. I got a lot of response. The following individuals helped with corrections, suggestions and material to improve this paper. They put in a big effort to help me get this document into its present shape. I would like to sincerely thank all of them. Naturally, all the mistakes you'll find in this book are mine. If you ever find a word which is spelled correctly, it must have been one of the people below dropping me a line.

> Rosemary Bailey, David Carlisle, Chris McCormack,
> David Dureisseix, Elliot, Robin Fairbairns,
> Alexandre Guimond, Neil Hammond, Rasmus Borup Hansen,
> Martien Hulsen, Eric Jacoboni, Alan Jeffrey, Byron Jones,
> David Jones, Andrzej Kawalec, Jörg Knappen, Maik Lehradt,
> Claus Malten, Hubert Partl, John Refling, Mike Ressler,
> Chris Rowley, Craig Schlenter, and Josef Tkadlec.

Preface

LaTeX[1] is a typesetting system which is most suited to producing scientific and mathematical documents of high typographical quality. The system is also suitable for producing all sorts of other documents, from simple letters to complete books. LaTeX uses TeX[2] as its formatting engine.

This short introduction describes LaTeX 2_ε and should be sufficient for most applications of LaTeX. For a complete description of the LaTeX system refer to [1, 3].

LaTeX is available for most computers from the IBM PC upwards. On many university computer networks the system is already installed, ready to operate. Information on how to access the local LaTeX installation should be provided in the *Local Guide* [4]. If you have problems getting started, ask the person who gave you this booklet. The scope of this document is *not* to tell you how to install and set up a LaTeX system, but to teach you how to write your documents so that they can be processed by LaTeX.

This Introduction is split into 4 chapters:

Chapter 1 tells you about the basic structure of LaTeX 2_ε documents. You will also learn a bit about the history of LaTeX. After reading this chapter you should have a rough picture of LaTeX. The picture will only be a framework, but it will enable you to integrate the information provided in the other chapters into the big picture.

Chapter 2 goes into the details of typesetting your documents. It explains most of the essential LaTeX commands and environments. After reading this chapter you will be able to write your first documents.

Chapter 3 explains how to typeset formulae with LaTeX. Again a lot of examples help you to understand how to use one of LaTeX main strengths. At the end of this chapter you find tables, listing all the mathematical symbols available in LaTeX.

Chapter 4 adds in some bits and bobs about LaTeX which are not essential, but very handy. Among other things you will learn how to include eps graphics into your documents or how to add an index to your publication.

It is important to read the chapters in sequential order. The book is not that big after all. Make sure to carefully read the examples, because a great part of the information is contained in the various examples you will find all through out the book.

If you need to get hold of any LaTeX related material, have a look in one of the `CTAN` ftp archives. For the US it is at `ftp.shsu.edu`, for Germany it is `ftp.dante.de` and for the UK it is `ftp.tex.ac.uk`. If you are not in one of these countries, choose the archive closest to you.

If you have ideas for something to be added, removed or altered in this document, please let me know. I am especially interested in feedback from LaTeX novices about which bits of this intro are easy to understand and which could be explained better.

Tobias Oetiker <oetiker@ee.ethz.ch>
Deptartment of Electrical Engineering, Swiss Federal Institute of Technology

The current version of this document will be available on
`CTAN:/tex-archive/info/lshort`

Contents

List of Figures

List of Tables

Chapter 1

Things You Need to Know

In the first part of this chapter you will get a short overview about the philosophy and history of LaTeX 2_ε. The second part of the chapter focuses on the basic structures of a LaTeX document. After reading this chapter you should have a rough knowledge of how LaTeX works. When reading on, this will help you, to integrate all the new information into the big picture.

1.1 The Name of the Game

1.1.1 TeX

TeX is a computer program by Donald E. Knuth [2]. It is aimed at typesetting text and mathematical formulae.

TeX is pronounced "Tech," with a "ch" as in the German word "Ach" or in Scottish "Loch." In an `ASCII` environment TeX becomes `TeX`.

1.1.2 LaTeX

LaTeX is a macro package which enables authors to typeset and print their work at the highest typographical quality, using a predefined, professional layout. LaTeX was originally written by Leslie Lamport [1]. It uses the TeX formatter as its typesetting engine.

Recently the LaTeX package has been updated by the LaTeX3 team, led by Frank Mittelbach, to include some long-requested improvements and to reunify all the patched versions which have cropped up since the release of LaTeX 2.09 some years ago. To distinguish the new version from the old, it is called LaTeX 2_ε. This documentation deals with LaTeX 2_ε.

LaTeX is pronounced "Lay-tech." If you refer to LaTeX in an `ASCII` environment you type `LaTeX`. LaTeX 2_ε is pronounced "Lay-tech two e" and typed `LaTeX2e`.

1.2 Basics

1.2.1 Author, Book Designer, and Typesetter

To publish something, authors give their typed manuscript to a publishing company. A book designer of the publishing company then decides the layout of the document (column width, fonts, space before and after headings, ...). The book designer writes his instructions into the manuscript and then gives it to a typesetter, who typesets the book according to these instructions.

A human book designer tries to find out what the author had in mind while writing the manuscript. He decides on chapter headings, citations, examples, formulae, etc. based on his professional knowledge and from the contents of the manuscript.

In a LaTeX environment, LaTeX takes the role of the book designer and uses TeX as its typesetter. But LaTeX is "only" a program and therefore needs more guidance. The author has to provide additional information which describes the logical structure of his work. This information is written into the text as "LaTeX commands."

This is quite different from the WYSIWYG[1] approach which most modern word processors such as *Word for Windows* or *WordPerfect* take. With these applications, authors specify the document layout interactively while typing text into the computer. All along the way, they can see on the screen how the final work will look when it is printed.

When using LaTeX it is normally not possible to see the final output while typing the text. But the final output can be previewed on the screen after processing the file with LaTeX. Then corrections can be made before actually sending the document to the printer.

1.2.2 Layout Design

Typographical design is a craft. Unskilled authors often commit serious formatting errors by assuming that book design is mostly a question of aesthetics—"If a document looks good artistically it is well designed." But as a document has to be read and not hung up in a picture gallery, the readability and understandability is of much greater importance than the beautiful look of it. Examples:

- The font size and numbering of headings have to be chosen to make the structure of chapters and sections clear to the reader.

- The line length has to be short enough that it does not strain the eyes of the reader, while long enough to fill the page beautifully.

[1] What you see is what you get.

With WYSIWYG systems, authors often generate aesthetically pleasing documents with very little or inconsistent structure. LaTeX prevents such formatting errors by forcing the author to declare the *logical* structure of his document. LaTeX then chooses the most suitable layout.

1.2.3 Advantages and Disadvantages

A topic often discussed when people from the WYSIWYG world meet LaTeX people, is "the advantages of LaTeX over a normal word processor" or the opposite. The best thing you can do when such a discussion starts, is to keep a low profile, as it often gets out of hand. But sometimes you cannot escape ...

The main advantages of LaTeX over normal word processors are the following:

- Professionally crafted layouts are available which make a document really look as if "printed."
- The typesetting of mathematical formulae is supported in a convenient way.
- The user only needs to learn a few easy to understand commands, which specify the logical structure of a document. They almost never need to tinker with the actual layout of the document.
- Even complex structures such as footnotes, references, table of contents, and bibliographies can be generated easily.
- For many typographical tasks not directly supported by basic LaTeX, there exist free add-on packages. For example, packages are available to include PostScript graphics or to typeset bibliographies conforming to exact standards. Many of these add-on packages are described in *The LaTeX Companion* [3].
- LaTeX encourages authors to write well structured texts because this is how LaTeX works—by specifying structure.
- TeX, the formatting engine of LaTeX 2$_\varepsilon$, is highly portable and free. Therefore the system runs on almost any hardware platform available.

LaTeX also has some disadvantages:

- More resources (memory, disk-space, computing power) are required to run a LaTeX system than a simple word processor. But things are getting better, as Word for Windows 6.0 needs even more disk space than a normal LaTeX system. When it comes down to processor usage, LaTeX beats any WYSIWYG system, as it only needs a lot of

CPU time when a document is actually processed, while WYSIWYG packages hog the CPU all the time.

- Although within a predefined document layout some parameters can be adjusted, the design of a whole new layout is difficult and takes a lot of time.[2]

1.3 LaTeX Input Files

The input for LaTeX is a plain `ASCII` text file. You can create it with any text editor. It contains the text of the document as well as the commands which tell LaTeX how to typeset the text.

1.3.1 Spaces

"Whitespace" characters such as blank or tab are treated uniformly as "space" by LaTeX. *Several consecutive* whitespace characters are treated as *one* "space". Whitespace at the start of a line is generally ignored and a single linebreak is treated like a "space".

An empty line between two lines of text defines the end of a paragraph. *Several* empty lines are treated the same as *one* empty line. The text below is an example. On the right hand side is the text from the input file and on the left hand side is the formatted output.

It does not matter whether you enter one or several spaces after a word.

An empty line starts a new paragraph.

```
It does not matter whether you
enter one or several     spaces
after a word.

An empty line starts a new
paragraph.
```

1.3.2 Special Characters

The following symbols are reserved characters, that either have a special meaning under LaTeX or are not available in all the fonts. If you enter them in your text directly, they will normally not print, but rather coerce LaTeX to do things you did not intend.

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

As you will see, these characters can be used in your documents all the same by adding a prefix backslash:

[2]Rumour says, that this is one of the key elements which will be addressed in the upcoming LaTeX3 system

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

The other symbols and many more can be printed with special commands in mathematical formulae or as accents.

1.3.3 LaTeX Commands

LaTeX commands are case sensitive and take one of the following two formats:

- They start with a backslash `\` and then have a name consisting only of letters. Command names are terminated by a space, a number or any other 'non-letter'.
- They consist of a backslash and exactly one special character.

LaTeX ignores whitespace after commands. If you want to get a space after a command, you have to put either `{}` and a blank or a special spacing command after the command name. The `{}` stops LaTeX from eating up all the space after the command name.

I read that Knuth divides the people working with TeX into TeXnicians and TeXperts.
Today is July 28, 1995.

```
I read that Knuth divides the
people working with \TeX{} into
\TeX{}nicians and \TeX perts.\\
Today is \today.
```

Some commands need a parameter which has to be given between curly braces `{ }` after the command name. Some commands support optional parameters which are added after the command name in square brackets `[ ]`. The next example uses some LaTeX commands. Don't worry about them, they will be explained later.

You can *lean* on me!

```
You can \textsl{lean} on me!
```

Please, start a new line right here!
Thank you!

```
Please, start a new line
right here!\linebreak[3]
Thank you!
```

1.3.4 Comments

When LaTeX encounters a `%` character while processing an input file, it ignores the rest of the present line. This is useful for adding notes to the input file, which will not show up in the printed version.

This is an example.

```
This is an % stupid
% Better: instructive <----
example.
```

1.4 Input File Structure

When LaTeX 2ε processes an input file it expects it to follow a certain structure. Thus every input file must start with the command

```
\documentclass{...}
```

This specifies what sort of document you intend to write. After that, you can include commands which influence the style of the whole document or you can load packages which add new features to the LaTeX system. To load such a package you use the command

```
\usepackage{...}
```

When all the setup work is done[3], you start the body of the text with the command

```
\begin{document}
```

Now you enter the text mixed with some useful LaTeX commands. At the end of the document you add the

```
\end{document}
```

command, which tells LaTeX to call it a day. Anything which follows this command will be ignored by LaTeX.

Figure 1.1 shows the contents of a minimal LaTeX 2ε file. A slightly more complicated input file is given in Figure 1.2.

[3]The area between `\documentclass` and `\begin{document}` is called *preamble*.

```
\documentclass{article}
\begin{document}
Small is beautiful.
\end{document}
```

Figure 1.1: A Minimal LaTeX File

1.5 The Layout of the Document

1.5.1 Document Classes

The first information LaTeX needs to know when processing an input file is the type of document the author wants to create. This is specified with the `\documentclass` command.

`\documentclass[`*options*`]{`*class*`}`

Here *class* specifies the type of document to be created. Table 1.1 lists the document classes explained in this introduction. The LaTeX 2_ε distribution provides additional classes for other documents including letters and slides. The *options* parameter customises the behaviour of the document class. The options have to be separated by commas. In Table 1.2 the most common options for the standard document classes are listed.

Example: An input file for a LaTeX document could start with the line

```
\documentclass[11pt,twoside,a4paper]{article}
```

it instructs LaTeX to typeset the document as an *article* with a base font size of *eleven points* and to produce a layout suitable for *double sided* printing on *a4 paper*.

1.5.2 Packages

While writing your document, you will probably find that there are some areas where basic LaTeX cannot solve your problem. If you want to include

```
\documentclass[a4paper,11pt]{article}
\usepackage{latexsym}
\author{H.~Partl}
\title{Minimalism}
\frenchspacing
\begin{document}
\maketitle
\tableofcontents
\section{Start}
Well and here begins my lovely article.
\section{End}
\ldots{} and here it ends.
\end{document}
```

Figure 1.2: Example of a Realistic Journal Article

Table 1.1: Document Classes

`article` for articles in scientific journals, presentations, short reports, program documentation, invitations, ...

`report` for longer reports containing several chapters, small books, PhD theses, ...

`book` for real books

`slide` for slides. The class uses big sans serife letters.

Table 1.2: Document Class Options

`10pt`, `11pt`, `12pt` Sets the size of the main font for the document. If no option is specified, `10pt` is assumed.

`a4paper`, `letterpaper`, ... Defines the paper size. The default size is `letterpaper`. Besides that, `a5paper`, `b5paper`, `executivepaper`, and `legalpaper` can be specified.

`fleqn` Typesets displayed formulae left-aligned instead of centred.

`leqno` Places the numbering of formulae on the left hand side instead of the right.

`titlepage`, `notitlepage` Specifies whether a new page should be started after the document title or not. The `article` class does not start a new page by default, while `report` and `book` do.

`twocolumn` Instructs LaTeX to typeset the document in two columns.

`twoside, oneside` Specifies whether double or single sided output should be generated. The classes `article` and `report` are single sided and the `book` class is double sided by default.

`openright, openany` Makes chapters begin either only on right hand pages or on the next page available. This does not work with the `article` class, as it does not know chapters. The `report` class by default starts chapters on the next page available and the `book` class starts them on right hand pages.

---

graphics, coloured text or source code from a file into your document, you need to enhance the capabilities of LaTeX. Such enhancements are called packages. Packages are activated with the

`\usepackage[`*options*`]{`*package*`}`

command. Where *package* is the name of the package and *options* is a list of keywords which trigger special features in the package. Some packages come with the LaTeX 2_ε base distribution (See Table 1.3). Others are provided separately. You may find more information on the packages installed at your site in your *Local Guide* [4]. The prime source for information about LaTeX is *The LaTeX Companion* [3]. It contains descriptions of hundreds of packages along with information of how to write your own extensions to LaTeX 2_ε.

Table 1.3: Some of the Packages Distributed with LaTeX

`doc` Allows the documentation of LaTeX programs.
Described in `doc.dtx` and in *The LaTeX Companion* [3].

`exscale` Provides scaled versions of the maths extension font.
Described in `ltexscale.dtx`.

`fontenc` Specifies which font encoding LaTeX should use.
Described in `ltoutenc.dtx`.

`ifthen` Provides commands of the form
'if...then do...otherwise do....'
Described in `ifthen.dtx` and *The LaTeX Companion* [3].

`latexsym` To access the LaTeX symbol font, you should use the `latexsym` package. Described in `latexsym.dtx` and in *The LaTeX Companion* [3]

`makeidx` Provides commands for producing indexes. Described in section 4.5 and in *The LaTeX Companion* [3].

`syntonly` Processes a document without typesetting it.
Described in `syntonly.dtx` and in *The LaTeX Companion* [3]. This is useful for quick error checking.

1.5.3 Page Styles

LaTeX supports three predefined header/footer combinations—so-called page styles. The *style* parameter of the

`\pagestyle{`*style*`}`

command defines which one to use. Table 1.4 lists the predefined page styles.

Table 1.4: The Predefined Page Styles of LaTeX

`plain`	prints the page numbers on the bottom of the page in the middle of the footer. This is the default page style.
`headings`	prints the current chapter heading and the page number in the header on each page while the footer remains empty. (This is the style used in this document)
`empty`	sets both the header and the footer to be empty.

It is possible to change the page style of the current page with the command

`\thispagestyle{`*style*`}`

In *The LaTeX Companion* [3] there is a description how to create your own headers and footers.

1.6 Big Projects

When working on big documents, you might want to split the input file into several parts. LaTeX has two commands which help you doing that.

`\include{`*filename*`}`

you can use this command in the document body, to insert the contents of another file. Note that LaTeX will start a new page before processing the material input from *filename*.

The second command can be used in the preamble. It allows you to instruct LaTeX to only input some of the `\include`d files.

`\includeonly{`*filename*`,`*filename*`,`...`}`

After this command is executed in the preamble of the document. Only `\include` commands for the filenames which are listed in the argument of the `\includeonly` command will be executed. Note that there is no space between the filename and the comma.

Chapter 2

Typesetting Text

After reading the previous chapter you should know about the basic stuff a LaTeX 2_ε document is made off. In this chapter I will fill in the remaining structure you will need to know in order to produce real world material.

2.1 Linebreaking and Pagebreaking

2.1.1 Justified Paragraphs

Often books are typeset with each line having the same length. LaTeX inserts the necessary linebreaks and spaces between words by optimising the contents of a whole paragraph. If necessary it also hyphenates words that would not fit comfortably on a line. How the paragraphs are typeset depends on the document class. Normally the first line of a paragraph is indented and there is no additional space between two paragraphs. Refer to section 4.2.2 for more information.

In special cases it might be necessary to order LaTeX to break a line:

`\\` or `\newline`

start a new line without starting a new paragraph.

`\\*`

additionally prohibits a pagebreak after the forced linebreak.

`\newpage`

starts a new page.

`\linebreak[`*n*`]`, `\nolinebreak[`*n*`]`, `\pagebreak[`*n*`]` and `\nopagebreak[`*n*`]`

do what their names say. They enable the author to influence their actions, with the optional argument n. It can be set to a number between zero to four. By setting n to a value below 4 you leave LaTeX the option of ignoring your command if the result would look very bad.

LaTeX always tries to produce the best linebreaks possible. If it cannot find a way to break the lines in a manner which meets its high standards, it rather lets one line stick out on the right of the paragraph. LaTeX then complains ("overfull hbox") while processing the input file. This happens most often when LaTeX cannot find a suitable place to hyphenate a word. By giving the `\sloppy` command you can instruct LaTeX to lower its standards a little. It then prevents such over-long lines by increasing the inter-word spacing — even if the final output is not optimal. In this case a warning ("underfull hbox") is given to the user. In most cases the result does not look too bad.

2.1.2 Hyphenation

LaTeX hyphenates words whenever necessary. If the hyphenation algorithm does not find the correct hyphenation points you can remedy the situation by using the following commands, to tell TeX about the exception.

The command

`\hyphenation{`*word list*`}`

causes the words listed in the argument to be hyphenated only at the points marked by "`-`". This command should be given in the preamble of the input file and should only contain words built from normal letters. The case of the letters is ignored. The example below will allow "hyphenation" to be hyphenated as well as "Hyphenation" and it prevents "FORTRAN", "Fortran" and "fortran" from being hyphenated at all. No accented characters or symbols are allowed in the argument.

Example:

```
\hyphenation{FORTRAN Hy-phen-a-tion}
```

The command `\-` inserts a discretionary hyphen into a word. This also becomes the only point hyphenation is allowed in this word. This command is especially useful for words containing special characters (eg. accented characters), because LaTeX does not automatically hyphenate words containing accented characters.

I think this is: supercalifragilisticex-
pialidocious

```
I think this is: su\-per\-cal\-%
i\-frag\-i\-lis\-tic\-ex\-pi\-%
al\-i\-do\-cious
```

Several words can be kept together on one line with the command

`\mbox{text}`

It causes its argument be kept together under all circumstances.

My phone number will change soon. It will be 0116 291 2319.

The parameter *filename* should contain the name of the file.

```
My phone number will change soon.
It will be \mbox{0116 291 2319}.

The parameter
\mbox{\emph{filename}} should
contain the name of the file.
```

2.2 Special Characters and Symbols

2.2.1 Quotation Marks

For quotation marks you should *not* use the `"` as on a typewriter. In publishing there are special opening and closing quotation marks. In LaTeX, use two `` ` ``s on for opening quotation marks and two `'`s for closing quotation marks.

“Please press the ‘x’ key.”

```
``Please press the `x' key.''
```

2.2.2 Dashes and Hyphens

LaTeX knows four kinds of dashes. You can access three of these with different numbers of consecutive dashes. The fourth kind is the mathematical minus:

daughter-in-law, X-rated
pages 13–67
yes—or no?
0, 1 and -1

```
daughter-in-law, X-rated\\
pages 13--67\\
yes---or no? \\
$0$, $1$ and $-1$
```

The names for these dashes are: `-` hyphen, `--` en-dash, `---` em-dash and `$-$` minus sign.

2.2.3 Ellipsis (. . .)

On a typewriter a comma or a period takes the same amount of space as any other letter. In book printing these characters occupy only a little space and are set very close to the preceding letter. Therefore you cannot enter "ellipsis" by just typing three dots, as the spacing would be wrong. Instead there is a special command for these dots. It is called

`\ldots`

Not like that ... but like that: New York, Tokyo, Budapest, …	`Not like that ... but like that:\\` `New York, Tokyo, Budapest, \ldots`

2.2.4 Ligatures

Some letter combinations are typeset not just by setting the different letters one after the other, but actually by using special symbols.

ff fi fl ffi… instead of f f f i f l f f i …

These so-called ligatures can be prohibited by inserting a `\mbox{}` between the two letters in question. This might be necessary with words built from two words.

Not shelfful but shelfful	`Not shelfful\\` `but shelf\mbox{}ful`

2.2.5 Accents and Special Characters

LaTeX supports the use of accents and special characters from many languages. Table 2.1 shows all sorts of accents being applied to the letter o. Naturally other letters work too.

To place an accent on top of an i or a j, their dots have to be removed. This is accomplished by typing `\i` and `\j`.

Hôtel, naïve, élève, smørrebrød, ¡Señorita!, Schönbrunner Schloß Straße	`H\^otel, na\"\i ve, \'el\`eve,\\` `sm\o rrebr\o d, !`Se\~norita!,\\` `Sch\"onbrunner Schlo\ss{}` `Stra\ss e`

2.3 International Language Support

If you need to write documents in languages other than English, LaTeX must apply different hyphenation rules in order to produce correct output.

For many languages, these changes can be accomplished by using the babel package by Johannes Braams. To use this package, your LaTeX system has to be specially configured. Your *Local Guide* [4] should give more information on this.

If your system is already appropriately configured, you can activate the babel package by adding the command

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

after the `\documentclass` command. Which *language*s your system supports should also be listed in the Local Guide.

For some languages babel also specifies new commands, which simplify the input of special characters. The German language for example, contains a lot of umlauts (äöü). With babel you can enter an ö by typing `"o` instead of `\"o`.

Some computer systems allow you to input special characters directly from the keyboard. LaTeX can handle such characters. Since the December 1995 release of LaTeX 2_ε, support for several input encodings is included in the basic distribution of LaTeX 2_ε. Check the inputenc package. When using this package you should consider that other people might not be able to display your input files on their computer, because they use a different encoding. For example, the German umlaut ä on a PC is encoded as 132 and on some Unix systems using ISO-LATIN 1 it is encoded as 228. Therefore, use this feature with care.

2.4 The Space between Words

To get a straight right margin in the output, LaTeX inserts varying amounts of space between the words. At the end of a sentence it inserts slightly more space, as this makes the text more readable. LaTeX assumes that sentences end with periods, question marks or exclamation marks. If a period follows an uppercase letter this is not taken as a sentence ending since periods after

Table 2.1: Accents and Special Characters

ò	\`o	ó	\'o	ô	\^o	õ	\~o
ō	\=o	ȯ	\.o	ö	\"o		
ŏ	\u o	ǒ	\v o	ő	\H o	o̧	\c o
ọ	\d o	o̱	\b o	o͡o	\t oo		
œ	\oe	Œ	\OE	æ	\ae	Æ	\AE
å	\aa	å	\aa	Å	\AA		
ø	\o	Ø	\O	ł	\l	Ł	\L
ı	\i	ȷ	\j	¡	!\`	¿	?\`

uppercase letters are normally for abbreviations.

Any exception from these assumptions has to be specified by the author. A backslash in front of a space generates a space which will not be enlarged. A tilde '~' character generates a space which cannot be enlarged and which additionally prohibits a linebreak. The command `\@` in front of a period specifies, that this period terminates a sentence even when it follows a uppercase letter.

Mr. Smith was happy to see her	`Mr.~Smith was happy to see her\\`
cf. Fig. 5	`cf.~Fig.~5\\`
I like BASIC. What about you?	`I like BASIC\@. What about you?`

The additional space after periods can be disabled with the command

```
\frenchspacing
```

which tells LaTeX *not* to insert any more space after a period than after ordinary character. This is very common in non-English languages. In this case the command `\@` is not necessary.

2.5 Titles, Chapters, and Sections

To help the reader find his or her way through your work, you should divide it into chapters, sections, and subsections. LaTeX supports this with special commands which take the section title as their argument. It is up to you to use them in the correct order.

For the `article` class the following sectioning commands are available:

```
\section{...}            \paragraph{...}
\subsection{...}         \subparagraph{...}
\subsubsection{...}      \appendix
```

For the `report` and the `book` class you can use two additional sectioning commands:

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

As the `article` class does not know about chapters, it is quite easy to add articles as chapters to a book. The spacing between sections, the numbering and the font size of the titles will be set automatically by LaTeX.

Two of the sectioning commands are a bit special:

- The `\part` command does not influence the numbering sequence of chapters.
- The `\appendix` command does not take an argument. It just changes the chapter[1] numbering to letters.

[1] For the article style it changes the section numbering

LaTeX creates a table of contents by taking the section headings and page numbers from the previous run of the document. The command

```
\tableofcontents
```

expands to a table of contents at the place where it is issued. A new document has to be processed ("LaTeXed") twice to get a correct table of contents. In some circumstances it might be necessary to compile the document a third time. LaTeX will tell you when this is necessary.

All sectioning commands listed above also exist as "starred" versions. A "starred" version of a command is built by adding a star `*` after the command name. They generate section headings which will not show up in the table of contents and which will not get numbered. The command `\section{Help}` for example would become `\section*{Help}`.

Normally the section headings show up in the table of contents exactly as they were entered in the text. Sometimes this is not possible, because the heading is too long to fit into the table of contents. The entry for the table of contents can therefore be specified as an optional argument before the actual heading.

```
\chapter[Read it! It's Exciting]{This is a very long
                     and especially boring title}
```

The title of the whole document is generated by issuing a

```
\maketitle
```

command. The contents of the title has to be defined by the commands

`\title{...}`, `\author{...}` and optionally `\date{...}`

before calling `\maketitle`. In the argument of `\authors` you can supply several names separated by `\and` commands.

An example of some of the above mentioned commands can be found in Figure 1.2 on page 7.

Apart from the sectioning commands explained above, LaTeX 2_ε introduced 3 additional commands, for use with the `book` class.

`\frontmatter`, `\mainmatter` and `\backmatter`

They are useful for dividing your publication. The commands alter chapter headings, and page numbering to work as you would expect it in a book.

2.6 Cross References

In books, reports and articles there are often cross references to figures, tables and special segments of text. LaTeX provides the following commands for cross referencing

`\label{`*marker*`}`, `\ref{`*marker*`}` and `\pageref{`*marker*`}`

Where *marker* is an identifier chosen by the user. LaTeX replaces **`\ref`** by the number of the section, subsection, figure, table, or theorem where the corresponding **`\label`** command was issued. **`\pageref`** prints the page number of the corresponding **`\label`** command. Here also the numbers from the previous run are used.

A reference to this subsection looks like: "see section 2.6 on page 20."

```
A reference to this subsection
\label{sec:this} looks like:
``see section~\ref{sec:this} on
page~\pageref{sec:this}.''
```

2.7 Footnotes

With the command

`\footnote{`*footnote text*`}`

a footnote will be printed at the foot of the current page.

Footnotes[a] are often used by people using LaTeX.

[a]This is a footnote

```
Footnotes\footnote{This
  is a footnote} are often used
by people using \LaTeX.
```

2.8 Emphasised Words

In manuscripts produced by typewriter, important words get underlined. In printed books these words are *emphasised.* The command to switch to an *emphasised* font is called

`\emph{`*text*`}`

Its argument is the text to be emphasised.

If you use emphasising *in an already emphasised text, then LaTeX uses an* upright *font for emphasising.*

```
\emph{If you use
  \emph{emphasising} in an
  already emphasised text, then
  \LaTeX{} uses an
  \emph{upright} font for
  emphasising.}
```

2.9 Environments

To typeset special purpose text, LaTeX defines many different environments for all sorts of formatting:

`\begin{`*name*`}` *text* `\end{`*name*`}`

Where *name* is the name of the environment. Environments can be called several times within each other as long as the calling order is maintained.

```
\begin{aaa}...\begin{bbb}...\end{bbb}...\end{aaa}
```

In the following sections all important environments are explained.

2.9.1 Itemise, Enumerate, and Description

The `itemize` environment is suitable for simple lists, the `enumerate` environment for enumerated lists, and the `description` environment for descriptions.

1. You can mix the list environments to your taste:
   - But it might start to look silly.
   - If you over-do it.
2. Therefore remember:

   Stupid things will not become smart because they are in a list.

   Smart things though, can be presented beautifully in a list.

```
\begin{enumerate}
\item You can mix the list
environments to your taste:
\begin{itemize}
\item But it might start to
look silly.
\item If you over-do it.
\end{itemize}
\item Therefore remember:
\begin{description}
\item[Stupid] things will not
become smart because they are
in a list.
\item[Smart] things though, can be
presented beautifully in a list.
\end{description}
\end{enumerate}
```

2.9.2 Flushleft, Flushright, and Center

The environments `flushleft` and `flushright` generate paragraphs which are either left or right aligned. The `center` environment generates centred text. If you do not issue `\\` to specify linebreaks, LaTeX will automatically determine linebreaks.

This text is
left aligned. LaTeX is not trying to
make each line the same length.

```
\begin{flushleft}
This text is\\ left aligned.
\LaTeX{} is not trying to make
each line the same length.
\end{flushleft}
```

This text is right
aligned. LaTeX is not trying to make
each line the same length.

```
\begin{flushright}
This text is right\\ aligned.
\LaTeX{} is not trying to make
each line the same length.
\end{flushright}
```

At the centre
of the earth

```
\begin{center}
At the centre\\of the earth
\end{center}
```

2.9.3 Quote, Quotation, and Verse

The `quote` environment is useful for quotes, important phrases and examples.

A typographical rule of thumb for the line length is:

> No line should contain more than 66 characters.

That's why multicolumn print is often used in newspapers.

```
A typographical rule of thumb
for the line length is:
\begin{quote}
No line should contain more than
66~characters.
\end{quote}
That's why multicolumn print is
often used in newspapers.
```

There are two similar environments: the `quotation` and the `verse` environments. The `quotation` environment is useful for longer quotes going over several paragraphs. The `verse` environment is useful for poems where the line breaks are important. The lines are separated by issuing a `\\` at the end of a line and a empty line after each verse.

I know only one English poem by heart. It is about Humpty Dumpty.

> Humpty Dumpty sat on a wall:
> Humpty Dumpty had a great fall.
> All the King's horses and all the King's men
> Couldn't put Humpty together again.

```
I know only one English poem by
heart. It is about Humpty Dumpty.
\begin{flushleft}
\begin{verse}
Humpty Dumpty sat on a wall:\\
Humpty Dumpty had a great fall.\\
All the King's horses and all
the King's men\\
Couldn't put Humpty together
again.
\end{verse}
\end{flushleft}
```

2.9.4 Printing Verbatim

Text which is enclosed between `\begin{verbatim}` and `\end{verbatim}` will be directly printed, as if it was typed on a typewriter, with all linebreaks and spaces, without any LaTeX command being executed.

Within a paragraph, similar functionality can be accessed with

`\verb+`*text*`+`

The `+` is just an example delimiter character. You can use any character except letters, `*` or blank. Many LaTeX examples in this booklet are typeset with this command.

The `\ldots` command …

```
The \verb|\ldots| command \ldots
```

```
10 PRINT "HELLO WORLD ";
20 GOTO 10
```

```
\begin{verbatim}
10 PRINT "HELLO WORLD ";
20 GOTO 10
\end{verbatim}
```

```
the␣starred␣version␣of
the␣␣␣␣␣␣verbatim
environment␣emphasises
the␣spaces␣␣␣in␣the␣text
```

```
\begin{verbatim*}
the starred version of
the      verbatim
environment emphasises
the spaces   in the text
\end{verbatim*}
```

The `\verb` command can be used in a similar fashion with a star:

```
like␣␣␣this␣:-)␣
```

```
\verb*|like   this :-) |
```

The `verbatim` environment and the `\verb` command may not be used within parameters of other commands.

2.9.5 Tabular

The `tabular` environment can be used to typeset beautiful tables with optional horizontal and vertical lines. LaTeX determines the width of the columns automatically.

The *table spec* argument of the

`\begin{tabular}{`*table spec*`}`

command defines the format of the table. Use an `l` for a column of left aligned text, `r` for right aligned text and `c` for centred text, `p{`*width*`}` for a column containing justified text with linebreaks, and `|` for a vertical line.

Within a `tabular` environment `&` jumps to the next column, `\\` starts a new line and `\hline` inserts an horizontal line.

7C0	hexadecimal
3700	octal
11111000000	binary
1984	decimal

```
\begin{tabular}{|r|l|}
\hline
7C0 & hexadecimal \\
3700 & octal \\
11111000000 & binary \\
\hline \hline
1984 & decimal \\
\hline
\end{tabular}
```

Welcome to Boxy's paragraph. We sincerely hope you'll all enjoy the show.

```
\begin{tabular}{|p{4.7cm}|}
\hline
Welcome to Boxy's paragraph.
We sincerely hope you'll
all enjoy the show.\\
\hline
\end{tabular}
```

With the `@{...}` construct it is possible to specify the column separator. This command kills the intercolumn space and replaces it with whatever is included in the curly braces. One common use for this command is explained below in the decimal alignment problem. Another possible usage is to suppress leading space in a table with `@{}`.

no leading space

```
\begin{tabular}{@{} l @{}}
\hline
no leading space\\\hline
\end{tabular}
```

leading space left and right

```
\begin{tabular}{l}
\hline
leading space left and right\\
\hline
\end{tabular}
```

Since there is no built-in way to align numeric columns on a decimal point[2], we can "cheat" and do it by using two columns: a right-aligned integer and a left-aligned fraction. The `@{}` command in the `\begin{tabular}` line replaces the normal intercolumn spacing with just a ".", giving the appearance of a single, decimal-point-justified column. Don't forget to replace the decimal point in your numbers with a column separator (`&`)! A column label can be placed above our numeric "column" by using the `\multicolumn` command.

Pi expression	Value
π	3.1416
π^{π}	36.46
$(\pi^{\pi})^{\pi}$	80662.7

```
\begin{tabular}{c r @{.} l}
Pi expression       &
\multicolumn{2}{c}{Value} \\
\hline
$\pi$               & 3&1416  \\
$\pi^{\pi}$         & 36&46   \\
$(\pi^{\pi})^{\pi}$ & 80662&7 \\
\end{tabular}
```

2.10 Floating Bodies

Today most publications contain a lot of figures and tables. These elements need special treatment because they cannot be broken across pages. One method would be to start a new page every time a figure or a table is too large to fit on the present page. This approach would leave pages partially empty which looks very bad.

The solution to this problem is to 'float' any figure or table, which does not fit on the current page, to a later page while filling the current page with body text. LaTeX offers two environments for floating bodies. One for tables and one for figures. To take full advantage of these two environments it is important to understand approximately how LaTeX handles floats internally. Otherwise floats may become a major source of frustration because LaTeX never puts them where you want them to be.

Let's first have a look at the commands LaTeX supplies for floats:

[2] If the 'tools' bundle is installed on your system, have a look at the dcolumn package

Any material enclosed in a `figure` or `table` environment will be treated as floating matter. Both float environments support an optional parameter

`\begin{figure}[`*placement specifier*`]` or `\begin{table}[`*placement specifier*`]`

called the *placement specifier*. This parameter is used to tell LaTeX about the locations the float is allowed to be moved to. A *placement specifier* is constructed by building a string of *float placing permissions*. See Table 2.2.

A table could be started with the following line e.g.

```
\begin{table}[!hbp]
```

The placement specifier `[!hbp]` allows LaTeX to place the table right here (`h`) or at the bottom (`b`) of some page or on a special floats page (`p`) and all that even if it does not look that good (`!`). If no placement specifier is given, the standard classes assume `[tbp]`.

LaTeX will place every float it encounters, according to the placement specifier supplied by the author. If a float cannot be placed on the current page it is deferred either to the *figures* or the *tables* queue[3]. When a new page is started, LaTeX first checks if it is possible to fill a special 'float' page with floats from the queues. If this is not possible, the first float on each queue is treated as if they had just occurred in the text: LaTeX tries again to place them according to their respective placement specifiers (except 'h' which is no longer possible). Any new floats occurring in the text get placed into the appropriate queues. LaTeX strictly maintains the original order of appearance for each type of float.

That's why a figure which cannot be placed, pushes all the further figures to the end of the document. Therefore:

> If LaTeX is not placing the floats as you expected, it is often only one float jamming one of the two float queues.

[3]These are fifo - 'first in first out' queues!

Table 2.2: Float Placing Permissions

Spec	Permission to place the float ...
`h`	*here* at the very place in the text where it occurred. This is useful mainly for small floats.
`t`	at the *top* of a page
`b`	at the *bottom* of a page
`p`	on a special *page* containing only floats.
`!`	without considering most of the internal parameters[a] which could stop this float from being placed.

[a]Such as the maximum number of floats allowed on one page

Having explained the difficult bit, there are some more things to mention about the `table` and `figure` environments. With the

`\caption{`*caption text*`}`

command you can define a caption for the float. A running number and the string "Figure" or "Table" will be added by LaTeX.

The two commands

`\listoffigures` and `\listoftables`

operate analogously to the `\tableofcontents` command, printing a list of figures or tables respectively. In these lists, the whole caption will be repeated. If you tend to use long captions, you must have a shorter version of the caption going into the lists. This is accomplished by entering the short version in brackets after the `\caption` command.

```
\caption[Short]{LLLLLoooooonnnnnggggg}
```

With `\label` and `\ref` you can create a reference to a float within your text.

The following example draws a square and inserts it into the document. You could use this if you wanted to reserve space for images you are going to paste into the finished document.

```
Figure~\ref{white} is an example of Pop-Art.
\begin{figure}[!hbp]
\makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
\caption{Five by Five in Centimetres} \label{white}
\end{figure}
```

In the example above[4] LaTeX will try *really hard* (`!`) to place the figure right *here* (`h`). If this is not possible, it tries to place the figure at the *bottom* (`b`) of the page. Failing to place the figure on the current page, it determines if it is possible to create a float page containing this figure and maybe some tables from the tables queue. If there is not enough material for a special float page, LaTeX starts a new page and once more treats the figure as if it had just occurred in the text.

Under certain circumstances it might be necessary to use the

`\clearpage`

command. It orders LaTeX to immediately place all floats remaining in the queues and then start a new page.

[4] assuming the figure queue is empty

Later in this introduction you will learn how to include PostScript drawings into your LaTeX 2ε documents.

2.11 Adding New Commands and Environments

In the first chapter, I explained that LaTeX needs information about the logical structure of the text to pick the right layout. This is a neat idea, but in practice you often get to the limits of this, because LaTeX just does not have a specialised environment or command for your exact purpose.

One solution is to use several LaTeX commands to produce the layout you have in mind. If you have to do this once, then there is no problem, but if it occurs repeatedly, it takes a lot of time. If you ever want to change the layout you have to work through the whole input file and edit all the items in question.

To solve this problem, LaTeX allows you to define your own commands and environments to the language.

2.11.1 New Commands

To add your own commands, use the

`\newcommand{`*name*`}[`*num*`]{`*definition*`}`

command. Basicly, the command requires two arguments. The *name* of the command you want to create and the *definition* of the command. The *num* argument in square brackets is optional. You can use it to create new commands which themselves take up to 9 arguments.

The following two examples should help you to get the idea. The first example defines a new command called `\tnss` this is short for "The Not So Short Introduction to LaTeX 2ε". Such a command could come in handy if you have to write the title of this book over and over again.

"The not so Short Introduction to LaTeX 2ε" … "The not so Short Introduction to LaTeX 2ε"

```
\newcommand{\tnss}
    {The not so Short Introduction
        to \LaTeXe}
% in the document body :
``\tnss'' \ldots{} ``\tnss''
```

The next example illustrates how to use the *num* argument. The `#1` sequence gets replace by the argument you specify. If you wanted to use more than one argument, use `#2` and so on.

- The *not so* Short Introduction to LaTeX 2ε
- The *very* Short Introduction to LaTeX 2ε

```
\newcommand{\txsit}[1]
    {The \emph{#1} Short
      Introduction to \LaTeXe}
% in the document body:
\begin{itemize}
\item \txsit{not so}
\item \txsit{very}
\end{itemize}
```

LaTeX will not allow you to create a new command which already exists. If you explicitly want to override an existing command you have to use the `\renewcommand`. Apart from its name, it uses the same syntax as the `\newcommand` command. In certain cases you might also want to use the `\providecommand` command. It works like `\newcommand`, but if the new command is already defined, LaTeX 2ε just ignores it.

2.11.2 New Environments

Analogously to the `\newcommand` command there is a command to create your own environments. When writing this introduction, I have created special environments for the structures I used repeatedly through out the introduction: "examples", "bits of code" and "command definition boxes". The `\newenvironment` command uses the following syntax:

`\newenvironment{`*name*`}[`*num*`]{`*before*`}{`*after*`}`

Like the `\newcommand` command, you can use `\newenvironment` with an optional argument and without. The material specified in the *before* argument is processed before the text in the environment gets processed. The material in the *after* argument gets processed when the `\end{`*name*`}` command is encountered.

The example below illustrates the usage of the `\newenvironment` command.

My humble subjects …

```
\newenvironment{king}
    {\begin{quote}}{\end{quote}}
% use this in the body part
\begin{king}
My humble subjects \ldots
\end{king}
```

The *num* argument is used the same way as in the `\newcommand` command. LaTeX makes sure that you do not define an environment which already exists. If you ever want to change an existing command, you can use the `\renewenvironment` command. It uses the same syntax as the `\newenvironment` command.

Chapter 3

Typesetting Mathematical Formulae

Now you are ready! In this chapter we will attack the main strength of TEX: mathematical typesetting. But be warned, this chapter only scratches the surface. While the things explained here are sufficient for many people, don't despair if you can't find a solution to your mathematical typesetting needs. It is highly likely that your problem is addressed in AMS-LATEX 2_ε [1] or some other package.

3.1 General

LATEX has a special mode for typesetting mathematics. Mathematical text within a paragraph is entered between `\(` and `\)`, between `$` and `$` or between `\begin{math}` and `\end{math}`.

Add a squared and b squared to get c squared. Or using a more mathematical approach: $c^{2}=a^{2}+b^{2}$

```
Add $a$ squared and $b$ squared
to get $c$ squared. Or using
a more mathematical approach:
$c^{2}=a^{2}+b^{2}$
```

TEX is pronounced as $\tau\epsilon\chi$.

100 m^{3} of water

This comes from my $\heartsuit$

```
\TeX{} is pronounced as
$\tau\epsilon\chi$.\\[6pt]
100~m$^{3}$ of water\\[6pt]
This comes from my $\heartsuit$
```

Larger mathematical equations or formulae are preferably typeset on separate lines. Therefore you enclose them between `\[` and `\]` or between `\begin{displaymath}` and `\end{displaymath}`. This produces formulae

[1] CTAN:/tex-archive/macros/latex/packages/amslatex

which are not enumerated. If you want LaTeX to enumerate them, you can use the `equation` environment.

Add a squared and b squared to get c squared. Or using a more mathematical approach:

$$c^{2}=a^{2}+b^{2}$$

And just one more line.

```
Add $a$ squared and $b$ squared
to get $c$ squared. Or using
a more mathematical approach:
\begin{displaymath}
c^{2}=a^{2}+b^{2}
\end{displaymath}
And just one more line.
```

With `\label` and `\ref` you can reference an equation within the text.

$$\epsilon > 0 \tag{3.1}$$

From (3.1) we gather ...

```
\begin{equation} \label{eq:eps}
\epsilon > 0
\end{equation}
From (\ref{eq:eps}) we gather
\ldots
```

There are differences between *math mode* and *text mode*. For example, in *math mode*:

1. Most spaces and linebreaks do not have any significance, as all spaces are either derived logically from the mathematical expressions or have to be specified using special commands such as `\,`, `\quad` or `\qquad`.

$$\forall x \in \mathbf{R}: \qquad x^{2} \geq 0 \tag{3.2}$$

```
\begin{equation}
\forall x \in \mathbf{R}:
\qquad x^{2} \geq 0
\end{equation}
```

2. Empty lines are not allowed. Only one paragraph per formula.

3. Each letter is considered to be the name of a variable and will be typeset as such. If you want to typeset normal text within a formula (normal upright font and normal spacing) then you have to enter the text using the `\textrm{...}` commands.

$$x^{2} \geq 0\qquad \textrm{for all }x\in\mathbf{R} \tag{3.3}$$

```
\begin{equation}
x^{2} \geq 0\qquad
\textrm{for all }x\in\mathbf{R}
\end{equation}
```

3.2 Grouping in Math Mode

Most math mode commands act only on the next character. So if you want several characters affected by a command you have to group them together using curly braces: `{...}`.

$$a^x + y \neq a^{x+y} \tag{3.4}$$

```
\begin{equation}
a^x+y \neq a^{x+y}
\end{equation}
```

3.3 Building Blocks of a Mathematical Formula

In this section the most important commands used in mathematical typesetting will be described. For a list of all symbols available take a look at section 3.9 on page 41.

Lowercase Greek letters are entered as `\alpha`, `\beta`, `\gamma`, ..., uppercase letters[2] are entered as `\Gamma`, `\Delta`, ...

$\lambda, \xi, \pi, \mu, \Phi, \Omega$

```
$\lambda,\xi,\pi,\mu,\Phi,\Omega$
```

Exponents and Subscripts can be specified using the `^` and the `_` character.

$a_1 \qquad x^2 \qquad e^{-\alpha t} \qquad a^3_{ij}$
$e^{x^2} \neq {e^x}^2$

```
$a_{1}$ \qquad $x^{2}$ \qquad
$e^{-\alpha t}$ \qquad
$a^{3}_{ij}$\\
$e^{x^2} \neq {e^x}^2$
```

The **square root sign** is entered as `\sqrt`, the n^{th} root is generated with `\sqrt[`n`]`. The size of the root sign is determined automatically by LaTeX.

$\sqrt{x} \qquad \sqrt{x^2+\sqrt{y}} \qquad \sqrt[3]{2}$

```
$\sqrt{x}$ \qquad
$\sqrt{ x^{2}+\sqrt{y} }$
\qquad $\sqrt[3]{2}$
```

The commands `\overline` and `\underline` create **horizontal lines** directly over or under an expression.

$\overline{m+n}$

```
$\overline{m+n}$
```

[2]There is no uppercase Alpha defined in LaTeX 2ε because it looks the same as a normal roman A. Once the new math coding is done, things will change.

The commands `\overbrace` and `\underbrace` create long **horizontal braces** over or under an expression.

$\underbrace{a+b+\cdots+z}_{26}$

```
$\underbrace{ a+b+\cdots+z }_{26}$
```

To add mathematical accents such as small arrows or tilde signs to variables you can use the commands given in table 3.1. Wide hats and tildes, covering several characters are generated with `\widetilde` and `\widehat`. With the `'` symbol you enter a dash.

$$y=x^{2}\qquad y'=2x\qquad y''=2$$

```
\begin{displaymath}
y=x^{2}\qquad y'=2x\qquad y''=2
\end{displaymath}
```

Often **vectors** are specified by adding small arrow symbols on top of a variable. This is done with the `\vec` command. To denote the vector from A to B the two commands `\overrightarrow` and `\overleftarrow` are useful.

$$\vec a\quad\overrightarrow{AB}$$

```
\begin{displaymath}
\vec a\quad\overrightarrow{AB}
\end{displaymath}
```

Names of log-like functions are often typeset in an upright font and not italic as variables. Therefore LaTeX supplies the following commands to typeset the most important function names:

`\arccos`	`\cos`	`\csc`	`\exp`	`\ker`	`\limsup`	`\min`	`\sinh`
`\arcsin`	`\cosh`	`\deg`	`\gcd`	`\lg`	`\ln`	`\Pr`	`\sup`
`\arctan`	`\cot`	`\det`	`\hom`	`\lim`	`\log`	`\sec`	`\tan`
`\arg`	`\coth`	`\dim`	`\inf`	`\liminf`	`\max`	`\sin`	`\tanh`

$$\lim_{n \rightarrow 0} \frac{\sin x}{x}=1$$

```
\[\lim_{n \rightarrow 0}
\frac{\sin x}{x}=1\]
```

For the modulo function there are two commands: `\bmod` for the binary operator "$a \bmod b$" and `\pmod` for expressions such as "$x\equiv a \pmod{b}$."

A built-up **fraction** is typeset with the `\frac{...}{...}` command. Often the slashed form 1/2 is preferable, because it looks better for small amounts of 'fraction material.'

$1\frac{1}{2}$ hours

```
$1\frac{1}{2}$~hours
```

$$\frac{x^{2}}{k+1}\qquad x^{\frac{2}{k+1}}\qquad x^{1/2}$$

```
\begin{displaymath}
\frac{ x^{2} }{ k+1 }\qquad
x^{ \frac{2}{k+1} }\qquad
x^{ 1/2 }
\end{displaymath}
```

To typeset binomial coefficients or similar structures you can use either the command `{... \choose ...}` or `{... \atop ...}`. The second command produces the same output as the first one, but without braces.

$${n \choose k}\qquad {x \atop y+2}$$

```
\begin{displaymath}
{n \choose k}\qquad {x \atop y+2}
\end{displaymath}
```

The **integral operator** is generated with `\int`, the **sum operator** with `\sum`. The upper and lower limits are specified with `^` and `_` as with subscripts and superscripts.

$$\sum_{i=1}^{n} \qquad \int_{0}^{\frac{\pi}{2}} \qquad$$

```
\begin{displaymath}
\sum_{i=1}^{n} \qquad
\int_{0}^{\frac{\pi}{2}} \qquad
\end{displaymath}
```

For **braces** and other delimiters there exist all types of symbols in TEX (e.g. $[\ \langle\ \|\ \updownarrow$). Round and square braces can be entered with the corresponding keys, curly braces with `\{`, all other delimiters are generated with special commands (eg. `\updownarrow`). For a list of all delimiters availble, check table 3.8 on page 43.

$${a,b,c}\neq\{a,b,c\}$$

```
\begin{displaymath}
{a,b,c}\neq\{a,b,c\}
\end{displaymath}
```

If you put the command `\left` in front of an opening delimiter or `\right` in front of a closing delimiter, TEX will automatically determine the correct size of the delimiter.

$$1 + \left(\frac{1}{ 1-x^{2} } \right) ^3$$

```
\begin{displaymath}
1 + \left( \frac{1}{ 1-x^{2} }
    \right) ^3
\end{displaymath}
```

In some cases it is necessary to specify the correct size of a mathematical delimiter by hand, therefore you can use the commands `\big`, `\Big`, `\bigg` and `\Bigg` as prefixes to most delimiter commands[3].

$$\Big((x+1)(x-1)\Big)^{2}$$
$$\big(\Big(\bigg(\Bigg(\quad \big\}\Big\}\bigg\}\Bigg\}\quad \big\|\Big\|\bigg\|\Bigg\|$$

```
$\Big( (x+1) (x-1) \Big) ^{2}$\\
$\big(\Big(\bigg(\Bigg($\quad
$\big\}\Big\}\bigg\}\Bigg\}$\quad
$\big\|\Big\|\bigg\|\Bigg\|$
```

To enter **three dots** into a formula you can use several commands. `\ldots` typesets the dots on the baseline, `\cdots` sets them centred. Beside that there are the commands `\vdots` for vertical and `\ddots` for diagonal dots. In section 3.5 you can find another example.

$$x_{1},\ldots,x_{n} \qquad x_{1}+\cdots+x_{n}$$

```
\begin{displaymath}
x_{1},\ldots,x_{n} \qquad
x_{1}+\cdots+x_{n}
\end{displaymath}
```

3.4 Math Spacing

If the spaces within formulae chosen by TeX are not satisfactory, they can be adjusted by inserting special spacing commands. The most important are: `\,` for a tiny space, `\␣` for a medium sized space (␣ stands for a "space" character), `\quad` and `\qquad` for large spaces and `\!` which shrinks a space.

$$\int\!\!\!\int_{D} dx\,dy \quad \textrm{instead of} \quad \int\int_{D} dx dy$$

```
\begin{displaymath}
\int\!\!\!\int_{D} dx\,dy \quad
\textrm{instead of} \quad
\int\int_{D} dx dy
\end{displaymath}
```

3.5 Vertically Aligned Material

To typeset **arrays**, use the `array` environment. It works somewhat similar to the `tabular` environment. The `\\` command is used to break the lines.

[3]These commands do not work as expected if a size changing command has been used, or the `11pt` or `12pt` option has been specified. Use the `exscale` or `amstex` packages to correct this behaviour

$$\mathbf{X} = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ddots \end{array} \right)$$

```
\begin{displaymath}
\mathbf{X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

The `array` environment can also be used to typeset which have one big delimiter by using a "`.`" as a invisible `\right` delimiter:

$$y = \left\{ \begin{array}{ll} a & \textrm{if } d>c \\ b+x & \textrm{in the morning} \\ l & \textrm{all day long} \end{array} \right.$$

```
\begin{displaymath}
y = \left\{ \begin{array}{ll}
 a & \textrm{if $d>c$}\\
 b+x & \textrm{in the morning}\\
 l & \textrm{all day long}
  \end{array} \right.
\end{displaymath}
```

For formulae running over several lines or for equation systems you can use the environments `eqnarray` and `eqnarray*` instead of `equation`. In `eqnarray` each line gets an equation number. In the `eqnarray*` no line numbers are produced.

The `eqnarray` and the `eqnarray*` environments work like a 3-column table of the form `{rcl}`, where the middle column can be used for the equal sign or the not-equal sign. The `\\` command breaks the lines.

$$f(x) = \cos x \quad (3.5)$$
$$f'(x) = -\sin x \quad (3.6)$$
$$\int_{0}^{x} f(y)dy = \sin x \quad (3.7)$$

```
\begin{eqnarray}
f(x) & = & \cos x       \\
f'(x) & = & -\sin x     \\
\int_{0}^{x} f(y)dy &
 = & \sin x
\end{eqnarray}
```

Long equations will not be automatically divided into neat bits. The author has to specify where to break them and how much to indent. The following two methods are the most common ones used to achieve this.

$$\begin{array}{rcl} \sin x & = & x - \frac{x^{3}}{3!} + \frac{x^{5}}{5!} - \\ & & {} - \frac{x^{7}}{7!} + \cdots \qquad (3.8) \end{array}$$

```
\begin{eqnarray}
\sin x & = & x -\frac{x^{3}}{3!}
     +\frac{x^{5}}{5!}-{}
                          \nonumber\\
 & & {}-\frac{x^{7}}{7!}+{}\cdots
\end{eqnarray}
```

$$\begin{aligned}\cos x = 1 - \frac{x^2}{2!} + \\ + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots \qquad (3.9)\end{aligned}$$

```
\begin{eqnarray}
\lefteqn{ \cos x = 1
     -\frac{x^{2}}{2!} +{} }
                   \nonumber\\
 & & {}+\frac{x^{4}}{4!}
     -\frac{x^{6}}{6!}+{}\cdots
\end{eqnarray}
```

The `\nonumber` command causes LaTeX not to generate a number for this equation.

3.6 Math Font Size

In math mode TeX selects the font size according to context. Superscripts for example get typeset in a smaller font. If you want to add roman text to an equation and use the `\textrm` command, the font size switching mechanism will not work, as `\textrm` temporarily escapes to text mode. Use `\mathrm` instead to keep the size switching mechanism active. But pay attention, `\mathrm` will only work well on short items. Spaces are still not active and accented characters do not work[4].

$$2^{\text{nd}} \quad 2^{\mathrm{nd}} \qquad (3.10)$$

```
\begin{equation}
2^\textrm{nd} \quad
2^\mathrm{nd}
\end{equation}
```

Sometimes you need to tell LaTeX about the correct font size nevertheless. In math mode the fontsize is set with the four commands:

`\displaystyle` $(\displaystyle 123)$, `\textstyle` $(\textstyle 123)$, `\scriptstyle` $(\scriptstyle 123)$ and `\scriptscriptstyle` $(\scriptscriptstyle 123)$.

$$\mathrm{corr}(X,Y)=\frac{\displaystyle\sum_{i=1}^n(x_i-\bar x)(y_i-\bar y)}{\displaystyle\sqrt{\sum_{i=1}^n(x_i-\bar x)^2\sum_{i=1}^n(y_i-\bar y)^2}}$$

```
\begin{displaymath}
\mathrm{corr}(X,Y)=
 \frac{\displaystyle
   \sum_{i=1}^n(x_i-\bar x)
   (y_i-\bar y)}
  {\displaystyle\sqrt{
 \sum_{i=1}^n(x_i-\bar x)^2
\sum_{i=1}^n(y_i-\bar y)^2}}
\end{displaymath}
```

[4]The AMS-LaTeX package makes the `\textrm` command work with size changing.

3.7 Describing Variables

For some formulae you might want to add a section after the formula where you describe the variables used in the expression. The following example should help you to get this done.

$$a^2 + b^2 = c^2$$

Where: a, b - are adjunct to the right angle of a right-angled triangle.

c - is the hypotenuse of the triangle.

```
\begin{displaymath}
a^2+b^2=c^2
\end{displaymath}
{\settowidth{\parindent}
   {Where:\ }

\makebox[0pt][r]
 {Where:\ }$a$, $b$ - are
adjunct to the right angle
of a right-angled triangle.

$c$ - is the hypotenuse of
the triangle.}
```

If you often need to typeset material like this, now the time is right to try out the `\newenvironment` command. Use it to create a specialised environment for describing variables. Check the description at the end of the previous chapter.

3.8 Theorems, Laws, ...

When writing mathematical documents, you probably need a way to typeset "Lemmas", "Definitions", "Axioms" and simmilar structures. LaTeX supports this with the command

`\newtheorem{`*name*`}[`*counter*`]{`*text*`}[`*section*`]`

The *name* argument is a short keyword used to identify the "theorem". Whith the *text* argument you define the actual name of the "theorem" which will be printed in the final document.

The arguments in square brackets are optional. They are both used to specify the numbering used on the "theorem". With the *counter* argument you can specify the *name* of a previously declared "theorem". The new "theorem" will then be numbered in the same sequence. The *section* argument allows you to specify the sectional unit within which you want your "theorem" to be numbered.

After executing the `\newtheorem` command in the preamble of your document, you can use the following command within the document.

> `\begin{`*name*`}[`*text*`]`
> This is my interesting theorem
> `\end{`*name*`}`

This should be enough theory. The following examples will hopefully remove the final remains of doubt, and make it clear that the `\newtheorem` environment is way to complex to understand.

Law 1 *Don't hide in the witness box*

Jury 2 (The Twelve) *It could be you! So beware and see law 1*

Law 3 *No, No, No*

```
% definitions for the document
% preamble
\newtheorem{law}{Law}
\newtheorem{jury}[law]{Jury}
%in the document
\begin{law} \label{law:box}
Don't hide in the witness box
\end{law}
\begin{jury}[The Twelve]
It could be you! So beware and
see law \ref{law:box}\end{jury}
\begin{law}No, No, No\end{law}
```

The "Jury" theorem uses the same counter as the "Law" theorem. Therefore, it gets a number which is in sequence with the other "Laws". The argument in square brackets is used to specify a title or something similar for the theorem.

Murphy 3.8.1 *What can go wrong, will go wrong.*

```
\newtheorem{mur}{Murphy}[section]
\begin{mur} What can go wrong,
will go wrong. \end{mur}
```

The "Murphy" theorem gets a number which is linked to the number of the current section. You could also use another unit, like chapter or subsection for example.

3.9 List of Mathematical Symbols

In the following tables you find all the symbols normally accessible from *math mode.*

To use the symbols listed in Tables 3.12–3.16[5], the package amssymb must be loaded in the preamble of the document and the AMS math fonts must be installed on the system. If the AMS package and fonts are not installed, on your system, have a look at
`CTAN:/tex-archive/macros/latex/packages/amslatex`

Table 3.1: Math Mode Accents

$\hat{a}$	`\hat{a}`	$\check{a}$	`\check{a}`	$\tilde{a}$	`\tilde{a}`	$\acute{a}$	`\acute{a}`
$\grave{a}$	`\grave{a}`	$\dot{a}$	`\dot{a}`	$\ddot{a}$	`\ddot{a}`	$\breve{a}$	`\breve{a}`
$\bar{a}$	`\bar{a}`	$\vec{a}$	`\vec{a}`	$\widehat{A}$	`\widehat{A}`	$\widetilde{A}$	`\widetilde{A}`

Table 3.2: Lowercase Greek Letters

α	`\alpha`	θ	`\theta`	o	`o`	υ	`\upsilon`
β	`\beta`	ϑ	`\vartheta`	π	`\pi`	ϕ	`\phi`
γ	`\gamma`	ι	`\iota`	ϖ	`\varpi`	φ	`\varphi`
δ	`\delta`	κ	`\kappa`	ρ	`\rho`	χ	`\chi`
ϵ	`\epsilon`	λ	`\lambda`	ϱ	`\varrho`	ψ	`\psi`
ε	`\varepsilon`	μ	`\mu`	σ	`\sigma`	ω	`\omega`
ζ	`\zeta`	ν	`\nu`	ς	`\varsigma`		
η	`\eta`	ξ	`\xi`	τ	`\tau`		

Table 3.3: Uppercase Greek Letters

Γ	`\Gamma`	Λ	`\Lambda`	Σ	`\Sigma`	Ψ	`\Psi`
Δ	`\Delta`	Ξ	`\Xi`	Υ	`\Upsilon`	Ω	`\Omega`
Θ	`\Theta`	Π	`\Pi`	Φ	`\Phi`		

[5]These tables were derived from `symbols.tex` by David Carlisle and subsequently changed extensivly as suggested by Josef Tkadlec

Table 3.4: Binary Relations

You can produce corresponding negations by adding a `\not` command as prefix to the following symbols.

$<$	`<`	$>$	`>`	$=$	`=`
$\leq$	`\leq` or `\le`	$\geq$	`\geq` or `\ge`	$\equiv$	`\equiv`
$\ll$	`\ll`	$\gg$	`\gg`	$\doteq$	`\doteq`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`
$\sqsubset$	`\sqsubset` [a]	$\sqsupset$	`\sqsupset` [a]	$\Join$	`\Join` [a]
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\bowtie$	`\bowtie`
$\in$	`\in`	$\ni$	`\ni` , `\owns`	$\propto$	`\propto`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\models$	`\models`
$\mid$	`\mid`	$\parallel$	`\parallel`	$\perp$	`\perp`
$\smile$	`\smile`	$\frown$	`\frown`	$\asymp$	`\asymp`
$:$	`:`	$\notin$	`\notin`	$\neq$	`\neq` or `\ne`

[a] Use the latexsym package to access this symbol

Table 3.5: Binary Operators

$+$	`+`	$-$	`-`		
$\pm$	`\pm`	$\mp$	`\mp`	$\triangleleft$	`\triangleleft`
$\cdot$	`\cdot`	$\div$	`\div`	$\triangleright$	`\triangleright`
$\times$	`\times`	$\setminus$	`\setminus`	$\star$	`\star`
$\cup$	`\cup`	$\cap$	`\cap`	$\ast$	`\ast`
$\sqcup$	`\sqcup`	$\sqcap$	`\sqcap`	$\circ$	`\circ`
$\vee$	`\vee` , `\lor`	$\wedge$	`\wedge` , `\land`	$\bullet$	`\bullet`
$\oplus$	`\oplus`	$\ominus$	`\ominus`	$\diamond$	`\diamond`
$\odot$	`\odot`	$\oslash$	`\oslash`	$\uplus$	`\uplus`
$\otimes$	`\otimes`	$\bigcirc$	`\bigcirc`	$\amalg$	`\amalg`
$\bigtriangleup$	`\bigtriangleup`	$\bigtriangledown$	`\bigtriangledown`	$\dagger$	`\dagger`
$\lhd$	`\lhd` [a]	$\rhd$	`\rhd` [a]	$\ddagger$	`\ddagger`
$\unlhd$	`\unlhd` [a]	$\unrhd$	`\unrhd` [a]	$\wr$	`\wr`

Table 3.6: BIG Operators

$\sum$	`\sum`	$\bigcup$	`\bigcup`	$\bigvee$	`\bigvee`	$\bigoplus$	`\bigoplus`
$\prod$	`\prod`	$\bigcap$	`\bigcap`	$\bigwedge$	`\bigwedge`	$\bigotimes$	`\bigotimes`
$\coprod$	`\coprod`	$\bigsqcup$	`\bigsqcup`			$\bigodot$	`\bigodot`
$\int$	`\int`	$\oint$	`\oint`			$\biguplus$	`\biguplus`

Table 3.7: Arrows

$\leftarrow$	`\leftarrow` or `\gets`	$\longleftarrow$	`\longleftarrow`	$\uparrow$	`\uparrow`
$\rightarrow$	`\rightarrow` or `\to`	$\longrightarrow$	`\longrightarrow`	$\downarrow$	`\downarrow`
$\leftrightarrow$	`\leftrightarrow`	$\longleftrightarrow$	`\longleftrightarrow`	$\updownarrow$	`\updownarrow`
$\Leftarrow$	`\Leftarrow`	$\Longleftarrow$	`\Longleftarrow`	$\Uparrow$	`\Uparrow`
$\Rightarrow$	`\Rightarrow`	$\Longrightarrow$	`\Longrightarrow`	$\Downarrow$	`\Downarrow`
$\Leftrightarrow$	`\Leftrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`	$\Updownarrow$	`\Updownarrow`
$\mapsto$	`\mapsto`	$\longmapsto$	`\longmapsto`	$\nearrow$	`\nearrow`
$\hookleftarrow$	`\hookleftarrow`	$\hookrightarrow$	`\hookrightarrow`	$\searrow$	`\searrow`
$\leftharpoonup$	`\leftharpoonup`	$\rightharpoonup$	`\rightharpoonup`	$\swarrow$	`\swarrow`
$\leftharpoondown$	`\leftharpoondown`	$\rightharpoondown$	`\rightharpoondown`	$\nwarrow$	`\nwarrow`
$\rightleftharpoons$	`\rightleftharpoons`	$\iff$	`\iff` (bigger spaces)	$\leadsto$	`\leadsto` [a]

[a]Use the latexsym package to access this symbol

Table 3.8: Delimiters

<table>
<tr><td>$($</td><td><code>(</code></td><td>$)$</td><td><code>)</code></td><td>$\uparrow$</td><td><code>\uparrow</code></td><td>$\Uparrow$</td><td><code>\Uparrow</code></td></tr>
<tr><td>$[$</td><td><code>[</code> or <code>\lbrack</code></td><td>$]$</td><td><code>]</code> or <code>\rbrack</code></td><td>$\downarrow$</td><td><code>\downarrow</code></td><td>$\Downarrow$</td><td><code>\Downarrow</code></td></tr>
<tr><td>$\{$</td><td><code>\{</code> or <code>\lbrace</code></td><td>$\}$</td><td><code>\}</code> or <code>\rbrace</code></td><td>$\updownarrow$</td><td><code>\updownarrow</code></td><td>$\Updownarrow$</td><td><code>\Updownarrow</code></td></tr>
<tr><td>$\langle$</td><td><code>\langle</code></td><td>$\rangle$</td><td><code>\rangle</code></td><td>$\vert$</td><td><code>|</code> or <code>\vert</code></td><td>$\Vert$</td><td><code>\|</code> or <code>\Vert</code></td></tr>
<tr><td>$\lfloor$</td><td><code>\lfloor</code></td><td>$\rfloor$</td><td><code>\rfloor</code></td><td>$\lceil$</td><td><code>\lceil</code></td><td>$\rceil$</td><td><code>\rceil</code></td></tr>
<tr><td>$/$</td><td><code>/</code></td><td>$\backslash$</td><td><code>\backslash</code></td><td></td><td>. (dual. empty)</td><td></td><td></td></tr>
</table>

Table 3.9: Large Delimiters

$\lgroup$	`\lgroup`	$\rgroup$	`\rgroup`	$\lmoustache$	`\lmoustache`	$\rmoustache$	`\rmoustache`
$\arrowvert$	`\arrowvert`	$\Arrowvert$	`\Arrowvert`	$\bracevert$	`\bracevert`		

Table 3.10: Miscellaneous Symbols

$\dots$	`\dots`	$\cdots$	`\cdots`	$\vdots$	`\vdots`	$\ddots$	`\ddots`
$\hbar$	`\hbar`	$\imath$	`\imath`	$\jmath$	`\jmath`	ℓ	`\ell`
$\Re$	`\Re`	$\Im$	`\Im`	$\aleph$	`\aleph`	$\wp$	`\wp`
$\forall$	`\forall`	$\exists$	`\exists`	$\mho$	`\mho` [a]	∂	`\partial`
$'$	`'`	$\prime$	`\prime`	$\emptyset$	`\emptyset`	∞	`\infty`
∇	`\nabla`	$\triangle$	`\triangle`	$\Box$	`\Box` [a]	$\Diamond$	`\Diamond` [a]
$\bot$	`\bot`	$\top$	`\top`	$\angle$	`\angle`	$\surd$	`\surd`
$\diamondsuit$	`\diamondsuit`	$\heartsuit$	`\heartsuit`	$\clubsuit$	`\clubsuit`	$\spadesuit$	`\spadesuit`
$\neg$	`\neg` or `\lnot`	$\flat$	`\flat`	$\natural$	`\natural`	$\sharp$	`\sharp`

[a] Use the latexsym package to access this symbol

Table 3.11: Non-Mathematical Symbols

These symbols can also be used in text mode.

†	`\dag`	§	`\S`	©	`\copyright`
‡	`\ddag`	¶	`\P`	£	`\pounds`

Table 3.12: AMS Delimiters

$\ulcorner$	`\ulcorner`	$\urcorner$	`\urcorner`	$\llcorner$	`\llcorner`	$\lrcorner$	`\lrcorner`

Table 3.13: AMS Greek and Hebrwe

$\digamma$	`\digamma`	$\varkappa$	`\varkappa`	$\beth$	`\beth`	$\daleth$	`\daleth`	$\gimel$	`\gimel`

Table 3.14: AMS Binary Relations

$\lessdot$	`\lessdot`	$\gtrdot$	`\gtrdot`	$\doteqdot$	`\doteqdot` or `\Doteq`
$\leqslant$	`\leqslant`	$\geqslant$	`\geqslant`	$\risingdotseq$	`\risingdotseq`
$\eqslantless$	`\eqslantless`	$\eqslantgtr$	`\eqslantgtr`	$\fallingdotseq$	`\fallingdotseq`
$\leqq$	`\leqq`	$\geqq$	`\geqq`	$\eqcirc$	`\eqcirc`
$\lll$	`\lll` or `\llless`	$\ggg$	`\ggg` or `\gggtr`	$\circeq$	`\circeq`
$\lesssim$	`\lesssim`	$\gtrsim$	`\gtrsim`	$\triangleq$	`\triangleq`
$\lessapprox$	`\lessapprox`	$\gtrapprox$	`\gtrapprox`	$\bumpeq$	`\bumpeq`
$\lessgtr$	`\lessgtr`	$\gtrless$	`\gtrless`	$\Bumpeq$	`\Bumpeq`
$\lesseqgtr$	`\lesseqgtr`	$\gtreqless$	`\gtreqless`	$\thicksim$	`\thicksim`
$\lesseqqgtr$	`\lesseqqgtr`	$\gtreqqless$	`\gtreqqless`	$\thickapprox$	`\thickapprox`
$\preccurlyeq$	`\preccurlyeq`	$\succcurlyeq$	`\succcurlyeq`	$\approxeq$	`\approxeq`
$\curlyeqprec$	`\curlyeqprec`	$\curlyeqsucc$	`\curlyeqsucc`	$\backsim$	`\backsim`
$\precsim$	`\precsim`	$\succsim$	`\succsim`	$\backsimeq$	`\backsimeq`
$\precapprox$	`\precapprox`	$\succapprox$	`\succapprox`	$\vDash$	`\vDash`
$\subseteqq$	`\subseteqq`	$\supseteqq$	`\supseteqq`	$\Vdash$	`\Vdash`
$\Subset$	`\Subset`	$\Supset$	`\Supset`	$\Vvdash$	`\Vvdash`
$\sqsubset$	`\sqsubset`	$\sqsupset$	`\sqsupset`	$\backepsilon$	`\backepsilon`
$\therefore$	`\therefore`	$\because$	`\because`	$\varpropto$	`\varpropto`
$\shortmid$	`\shortmid`	$\shortparallel$	`\shortparallel`	$\between$	`\between`
$\smallsmile$	`\smallsmile`	$\smallfrown$	`\smallfrown`	$\pitchfork$	`\pitchfork`
$\vartriangleleft$	`\vartriangleleft`	$\vartriangleright$	`\vartriangleright`	$\blacktriangleleft$	`\blacktriangleleft`
$\trianglelefteq$	`\trianglelefteq`	$\trianglerighteq$	`\trianglerighteq`	$\blacktriangleright$	`\blacktriangleright`

Table 3.15: AMS Arrows

$\dashleftarrow$	`\dashleftarrow`	$\dashrightarrow$	`\dashrightarrow`	$\multimap$	`\multimap`
$\leftleftarrows$	`\leftleftarrows`	$\rightrightarrows$	`\rightrightarrows`	$\upuparrows$	`\upuparrows`
$\leftrightarrows$	`\leftrightarrows`	$\rightleftarrows$	`\rightleftarrows`	$\downdownarrows$	`\downdownarrows`
$\Lleftarrow$	`\Lleftarrow`	$\Rrightarrow$	`\Rrightarrow`	$\upharpoonleft$	`\upharpoonleft`
$\twoheadleftarrow$	`\twoheadleftarrow`	$\twoheadrightarrow$	`\twoheadrightarrow`	$\upharpoonright$	`\upharpoonright`
$\leftarrowtail$	`\leftarrowtail`	$\rightarrowtail$	`\rightarrowtail`	$\downharpoonleft$	`\downharpoonleft`
$\leftrightharpoons$	`\leftrightharpoons`	$\rightleftharpoons$	`\rightleftharpoons`	$\downharpoonright$	`\downharpoonright`
$\Lsh$	`\Lsh`	$\Rsh$	`\Rsh`	$\rightsquigarrow$	`\rightsquigarrow`
$\looparrowleft$	`\looparrowleft`	$\looparrowright$	`\looparrowright`	$\leftrightsquigarrow$	`\leftrightsquigarrow`
$\curvearrowleft$	`\curvearrowleft`	$\curvearrowright$	`\curvearrowright`		
$\circlearrowleft$	`\circlearrowleft`	$\circlearrowright$	`\circlearrowright`		

Table 3.16: AMS Negated Binary Relations and Arrows

$\nless$	\nless	$\ngtr$	\ngtr	$\varsubsetneqq$	\varsubsetneqq
$\lneq$	\lneq	$\gneq$	\gneq	$\varsupsetneqq$	\varsupsetneqq
$\nleq$	\nleq	$\ngeq$	\ngeq	$\nsubseteqq$	\nsubseteqq
$\nleqslant$	\nleqslant	$\ngeqslant$	\ngeqslant	$\nsupseteqq$	\nsupseteqq
$\lneqq$	\lneqq	$\gneqq$	\gneqq	$\nmid$	\nmid
$\lvertneqq$	\lvertneqq	$\gvertneqq$	\gvertneqq	$\nparallel$	\nparallel
$\nleqq$	\nleqq	$\ngeqq$	\ngeqq	$\nshortmid$	\nshortmid
$\lnsim$	\lnsim	$\gnsim$	\gnsim	$\nshortparallel$	\nshortparallel
$\lnapprox$	\lnapprox	$\gnapprox$	\gnapprox	$\nsim$	\nsim
$\nprec$	\nprec	$\nsucc$	\nsucc	$\ncong$	\ncong
$\npreceq$	\npreceq	$\nsucceq$	\nsucceq	$\nvdash$	\nvdash
$\precneqq$	\precneqq	$\succneqq$	\succneqq	$\nvDash$	\nvDash
$\precnsim$	\precnsim	$\succnsim$	\succnsim	$\nVdash$	\nVdash
$\precnapprox$	\precnapprox	$\succnapprox$	\succnapprox	$\nVDash$	\nVDash
$\subsetneq$	\subsetneq	$\supsetneq$	\supsetneq	$\ntriangleleft$	\ntriangleleft
$\varsubsetneq$	\varsubsetneq	$\varsupsetneq$	\varsupsetneq	$\ntriangleright$	\ntriangleright
$\nsubseteq$	\nsubseteq	$\nsupseteq$	\nsupseteq	$\ntrianglelefteq$	\ntrianglelefteq
$\subsetneqq$	\subsetneqq	$\supsetneqq$	\supsetneqq	$\ntrianglerighteq$	\ntrianglerighteq
$\nleftarrow$	\nleftarrow	$\nrightarrow$	\nrightarrow	$\nleftrightarrow$	\nleftrightarrow
$\nLeftarrow$	\nLeftarrow	$\nRightarrow$	\nRightarrow	$\nLeftrightarrow$	\nLeftrightarrow

Table 3.17: AMS Binary Operators

$\dotplus$	\dotplus	$\centerdot$	\centerdot	$\intercal$	\intercal
$\ltimes$	\ltimes	$\rtimes$	\rtimes	$\divideontimes$	\divideontimes
$\Cup$	\Cup or \doublecup	$\Cap$	\Cap or \doublecap	$\smallsetminus$	\smallsetminus
$\veebar$	\veebar	$\barwedge$	\barwedge	$\doublebarwedge$	\doublebarwedge
$\boxplus$	\boxplus	$\boxminus$	\boxminus	$\circleddash$	\circleddash
$\boxtimes$	\boxtimes	$\boxdot$	\boxdot	$\circledcirc$	\circledcirc
$\leftthreetimes$	\leftthreetimes	$\rightthreetimes$	\rightthreetimes	$\circledast$	\circledast
$\curlyvee$	\curlyvee	$\curlywedge$	\curlywedge		

Table 3.18: AMS Miscellaneous

$\hbar$	`\hbar`	$\hslash$	`\hslash`	$\Bbbk$	`\Bbbk`
$\square$	`\square`	$\blacksquare$	`\blacksquare`	$\circledS$	`\circledS`
$\vartriangle$	`\vartriangle`	$\blacktriangle$	`\blacktriangle`	$\complement$	`\complement`
$\triangledown$	`\triangledown`	$\blacktriangledown$	`\blacktriangledown`	$\Game$	`\Game`
$\lozenge$	`\lozenge`	$\blacklozenge$	`\blacklozenge`	$\bigstar$	`\bigstar`
$\angle$	`\angle`	$\measuredangle$	`\measuredangle`	$\sphericalangle$	`\sphericalangle`
$\diagup$	`\diagup`	$\diagdown$	`\diagdown`	$\backprime$	`\backprime`
$\nexists$	`\nexists`	$\Finv$	`\Finv`	$\varnothing$	`\varnothing`
$\eth$	`\eth`	$\mho$	`\mho`		

Table 3.19: Math Alphabets

Example	Command	Required package
ABCdef	`\mathrm{ABCdef}`	
ABCdef	`\mathit{ABCdef}`	
$\mathnormal{ABCdef}$	`\mathnormal{ABCdef}`	
$\mathcal{ABC}$	`\mathcal{ABC}`	
$\mathcal{ABC}$	`\mathcal{ABC}`	euscript with with option: mathcal
	`\mathscr{ABC}`	euscript with option: mathscr
$\mathfrak{ABCdef}$	`\mathfrak{ABCdef}`	eufrak
$\mathbb{ABC}$	`\mathbb{ABC}`	amsfonts or amssymb

Chapter 4

Specialities

Don't read this chapter! — Or, at least, if you feel confident enough, you can now start writing your documents in LaTeX. The purpose of this chapter is to add some 'spice' to your LaTeX knowledge. A much more complete description of specialities and enhancements possible with LaTeX can be found in the *LaTeX Manual* [1] and *The LaTeX Companion* [3].

4.1 Fonts and Sizes

LaTeX chooses the appropriate font and font size based on the logical structure of the document (sections, footnotes, ...). In some cases one might like to change fonts and sizes by hand. To do this you can use the commands listed in Tables 4.1 and 4.2.

The small and **bold** Romans ruled all of great big *Italy*.

```
{\small The small and
\textbf{bold} Romans ruled}
{\Large all of great big
\textit{Italy}.}
```

In *math mode* you can use the font changing *commands* to temporarily exit *math mode* and enter some normal text. If you want to switch to another font for math typesetting there exists another special set of commands. Refer to Table 4.3.

In connection with the font size commands, curly braces play a significant role. They are used to to build *groups*. Groups limit the scope of most LaTeX commands.

He likes large and small letters.

```
He likes {\LARGE large and
{\small small} letters}.
```

The font size commands also change the line spacing, but only if the

Table 4.1: Fonts

`\textrm{...}`	roman	`\textsf{...}`	sans serif
`\texttt{...}`	`typewriter`		
`\textmd{...}`	medium	`\textbf{...}`	**bold face**
`\textup{...}`	upright	`\textit{...}`	*italic*
`\textsl{...}`	*slanted*	`\textsc{...}`	SMALL CAPS
`\emph{...}`	*emphasised*	`\textnormal{...}`	document font

Table 4.2: Font sizes

`\tiny`	tiny font	`\Large`	larger font
`\scriptsize`	very small font	`\LARGE`	very large font
`\footnotesize`	quite small font	`\huge`	huge
`\small`	small font	`\Huge`	largest
`\normalsize`	normal font		
`\large`	large font		

Table 4.3: Math fonts

Command	*Example*	*Output*
`\mathcal{...}`	`$\mathcal{B}=c$`	$\mathcal{B}=c$
`\mathrm{...}`	`$\mathrm{K}_2$`	K_2
`\mathbf{...}`	`$\sum x=\mathbf{v}$`	$\sum x=\mathbf{v}$
`\mathsf{...}`	`$\mathsf{G\times R}$`	$\mathsf{G\times R}$
`\mathtt{...}`	`$\mathtt{L}(b,c)$`	$\mathtt{L}(b,c)$
`\mathnormal{...}`	`$\mathnormal{R_1}=R_1$`	$\mathnormal{R_1}=R_1$
`\mathit{...}`	`$finder\neq\mathit{finder}$`	$finder\neq\mathit{finder}$

paragraph ends within the scope of the font size command. The closing curly brace `}` should therefore not come too early. Note the position of the `\par` command in the next two examples.

Don't read this! It is not true. You can believe me!

```
{\Large Don't read this! It is not
true. You can believe me!\par}
```

This is not true. But remember I am liar.

```
{\Large This is not true.
But remember I am liar.}\par
```

To conclude this journey into the land of fonts and font sizes, here is a little word of advice[1]:

> **Remember!** *The* MO RE fonts you use **in** a document *the* more READABLE and *beautiful it becomeS*.

4.2 Spacing

4.2.1 Line Spacing

If you want to use larger inter line spacing in a document, you can change its value by putting the

`\linespread{`*factor*`}`

command into the preamble of your document. Use `\linespread{1.3}` for "one and a half" line spacing, and `\linespread{1.6}` for "double" line spacing. Normally the lines are not spread, therefore the default line spread factor is 1.

4.2.2 Paragraph Formating

In LaTeX, there are two parameters influencing paragraph layout. By placing a definition like

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

in the preamble of the input file[2] the appearance of paragraphs can be changed. These two lines increase the space between two paragraphs while setting the paragraph indent to zero. In continental Europe, paragraphs are often separated by some space and not indented.

[1] Attention: This is a bit of satire. I hope you realise that!

[2] Between the `\documentclass` and the `\begin{document}` commands

If you want to indent a paragraph which is not indented, you can use

```
\indent
```

at the beginning of the paragraph[3]. Obviously this can only work, when `\parindent` is not set to zero.

To create a non indented paragraph you can use

```
\noindent
```

as the first command of the paragraph. This might come in handy, when you start a document with body text and not with a sectioning command.

4.2.3 Horizontal Space

LaTeX determines the spaces between words and sentences automatically. To add horizontal space, use:

`\hspace{`*length*`}`

If such a space should be kept, even if it falls at the end or the start of a line, use `\hspace*` instead of `\hspace`. The *length* in the simplest case just is 'a number plus a unit. The most important units are listed in Table 4.4.

This   is a space of 1.5 cm.

```
This\hspace{1.5cm}is a space
of 1.5 cm.
```

The command

`\stretch{`n`}`

generates a special rubber space. It stretches, until all the remaining space on a line is filled up. If two `\hspace{\stretch{`n`}}` commands are issued on the same line, they grow according to the stretch factor.

x   x   x

```
x\hspace{\stretch{1}}
x\hspace{\stretch{3}}x
```

[3]To indent the first paragraph after each section head, use the indentfirst package in the 'tools' bundle

Table 4.4: TEX Units

`mm`	millimeter ≈ 1/25 inch ⊔
`cm`	centimeter = 10 mm ⊔
`in`	inch ≈ 25 mm ⊔
`pt`	point ≈ 1/72 inch ≈ $\frac{1}{3}$ mm ⊔
`em`	approx width of an `m` in the current font ⊔
`ex`	approx height of an `x` in the current font ⊔

4.2.4 Vertical Space

The space between paragraphs, sections, subsections, ... is determined automatically by LATEX. If necessary, additional vertical space *between two paragraphs* can be added with the command:

`\vspace{`*length*`}`

This command should normally be used between two empty lines. If the space should be preserved at the top, or at the bottom of a page, use the starred version of the command `\vspace*` instead of `\vspace`.

The `\stretch` command in connection with `\pagebreak` can be used to typeset text on the last line of a page, or to centre text vertically on a page.

```
Some text \ldots

\vspace{\stretch{1}}
This goes onto the last line of the page.\pagebreak
```

Additional space between two lines of *the same* paragraph or within a table is specified with the

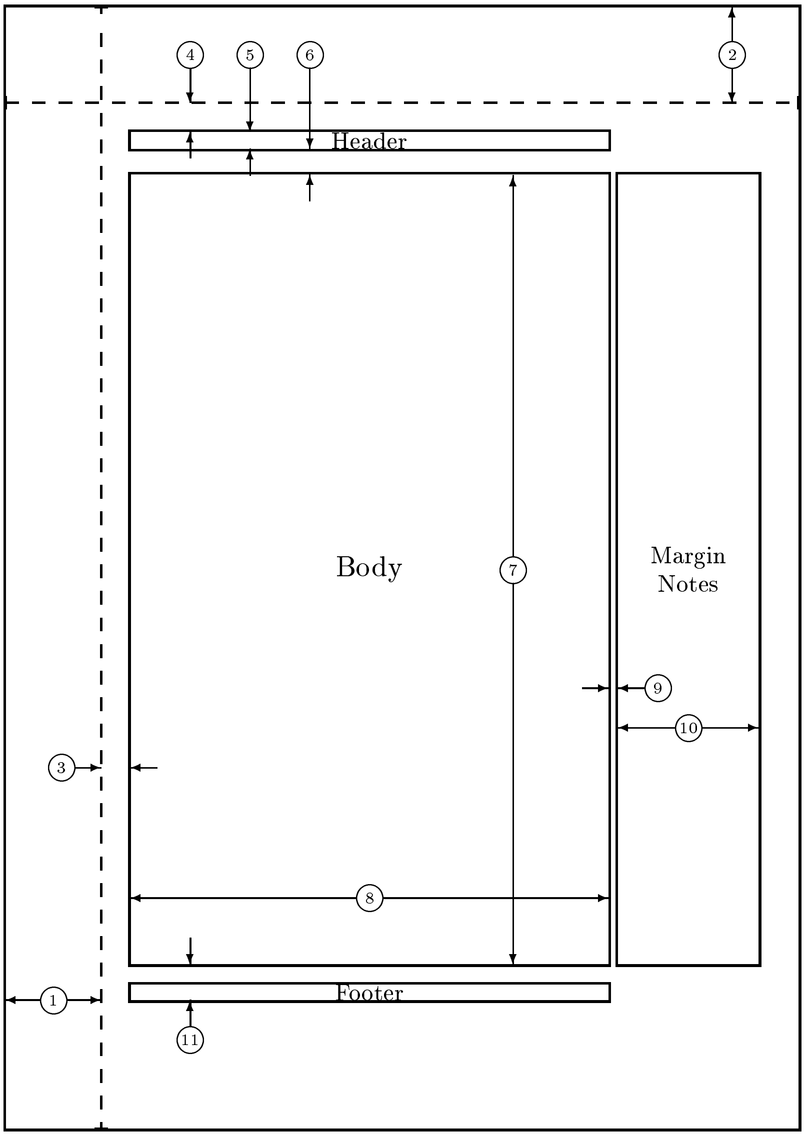
`\\[`*length*`]`

command.

4.3 Page Layout

LATEX 2_ε allows you to specify the paper size in the `\documentclass` command. It then automatically picks the right text margins. But some times, you may not be happy with the predefined values. Naturally, you can change them. Figure 4.1 shows all the parameters which can be changed. The figure was produced with the layout package from the tools bundle[4].

[4] `CTAN:/tex-archive/macros/latex/packages/tools`



1	`one inch + \hoffset`	2	`one inch + \voffset`
3	`\oddsidemargin = 22pt`	4	`\topmargin = 22pt`
5	`\headheight = 13pt`	6	`\headsep = 19pt`
7	`\textheight = 595pt`	8	`\textwidth = 360pt`
9	`\marginparsep = 7pt`	10	`\marginparwidth = 106pt`
11	`\footskip = 27pt`		`\marginparpush = 5pt (not shown)`
	`\hoffset = 0pt`		`\voffset = 0pt`
	`\paperwidth = 597pt`		`\paperheight = 845pt`

Figure 4.1: Page Layout Parameters

LaTeX provides two commands to change these parameters. They are usually used in the document preamble.

The first command assigns a fixed value to any of the parameters:

`\setlength{`*parameter*`}{`*length*`}`

The second command, adds a length to any of the parameters.

`\addtolength{`*parameter*`}{`*length*`}`

This second command is actually more useful than the `\setlength` command, because you can now work relative to the existing settings. To add one centimetre to the overall text width, I would put the following commands into the document preamble:

```
\addtolength{\hoffset}{-0.5cm}
\addtolength{\textwidth}{1cm}
```

4.4 Bibliography

With the `thebibliography` environment you can produce a bibliography. Each entry starts with

`\bibitem{`*marker*`}`

The *marker* is then used to cite the book within the document.

`\cite{`*marker*`}`

The numbering of the entries is generated automatically. The parameter after the `\begin{thebibliography}` command sets the maximum width of these numbers.

Partl [1] has proposed, that …

```
Partl~\cite{pa} has
proposed, that \ldots

\begin{thebibliography}{99}
\bibitem{pa} H.~Partl:
\emph{German \TeX},
TUGboat Vol.~9, No.~1 ('88)
\end{thebibliography}
```

Bibliography

[1] H. Partl: *German TEX*, TUGboat Vol. 9, No. 1 ('88)

4.5 Indexing

A very useful feature of many books is their index. With LATEX and the support program `makeindx`[5] indexes can be generated quite easily. In this introduction, only the basic index generation commands will be explained. For a more in depth view please refer to *The LATEX Companion* [3].

To enable the indexing feature of LATEX the makeidx package must be loaded in the preamble with:

```
\usepackage{makeidx}
```

and the special indexing commands must be enabled by putting the

```
\makeindex
```

command into the input files preamble.

The contents of the index is specified with

```
\index{key}
```

commands. Where *key* is the index entry. You enter the index commands at the points in the text where you want the final index entries to point to. Table 4.5 explains the syntax of the *key* argument with several examples.

When the input file is processed with LATEX, each `\index` command writes an appropriate index entry together with the current page number to a special file. The file has the same name as the LATEX input file, but a

[5] On systems supporting filenames longer than 8 characters, the program may be called `makeindex`.

Table 4.5: Index Key Syntax Examples

Example	**Index Entry**	**Comment**
`\index{hello}`	hello, 1	Plain entry
`\index{hello!Peter}`	Peter, 3	Subentry under 'hello'
`\index{Sam@\textsl{Sam}}`	*Sam*, 2	Formated entry
`\index{Lin@\textbf{Lin}}`	**Lin**, 7	Same as above
`\index{Jenny|textbf}`	Jenny, **3**	Formated page number
`\index{Joe|textit}`	Joe, *5*	Same as above

different extension (`.ind`). This `.ind` file can then be processed with the `makeindx` program.

```
makeidx filename
```

The `makeindx` program generates a sorted index with the same base file name, but this time with the extension `.idx`. If now the LaTeX input file is processed again, this sorted index gets included into the document at the point where LaTeX finds

```
\printindex
```

The showidx package which comes with LaTeX 2_ε prints out all index entries in the left margin of the text. This is quite useful for proofreading a document and verifying the index.

4.6 Including EPS Graphics

With the `figure` and the `table` environment LaTeX provides the basic facilities to work with floating bodies such as images or graphics.

There are also several possibilities to generate the actual graphics with basic LaTeX or a LaTeX extension package. Unfortunately, most users find them quite difficult to understand. Therefore this will not be explained any further in this manual. For more information on that subject please refer to *The LaTeX Companion* [3] and the *LaTeX Manual* [1].

A much easier way to get graphics into a document, is to generate them with a specialised software package[6] and then include the finished graphics into the document. Here again, LaTeX packages offer many ways to do that. In this introduction, only the use of Encapsulated PostScript (EPS) graphics will be discussed, because it is quite easy to do and widely used. In order

[6]Such as XFig, CorelDraw!, Freehand, GNU Plot, ...

to use pictures in the EPS format, you must have a PostScript printer[7] available for output.

A good set of commands for inclusion of graphics is provided in the graphicx package by D. P. Carlisle. It is part of a whole family of packages called the "graphics" bundle[8].

Assuming you are working on a system with a PostScript printer available for output and with the graphicx package installed, you can use the following step by step guide to include a picture into your document:

1. Export the picture from your graphics program in EPS format.

2. Load the graphicx package in the preamble of the input file with

   `\usepackage[`*driver*`]{graphicx}`

   driver is the name of your "dvi to postscript" converter[9]. This information is required by the package because the actual graphics inclusion is done by the printer driver. Knowing the *driver*, the graphicx package inserts the correct commands into the `.dvi` file for the printer driver to include the desired EPS graphics.

3. Use the command

   `\includegraphics[`*key*=*value*, ...`]{`*file*`}`

   to include *file* into your document. The optional parameter accepts a comma separated list of *keys* and associated *values*. The *keys* can be used to alter the width, height and rotation of the included graphic. Table 4.6 lists the most important keys.

Table 4.6: Key Names for graphicx Package

`width`	scale graphic to the specified width
`height`	scale graphic to the specified height
`angle`	rotate graphics clockwise

The following example code will hopefully make things clear:

[7]Another possibility to output PostScript is the GHOSTSCRIPT program available from `CTAN:/tex-archive/support/ghostscript`

[8]`CTAN:/tex-archive/macros/latex/packages/graphics`

[9]The most widely used program is called `dvips`.

```
\begin{figure}
\begin{center}
\includegraphics[angle=90, width=10cm]{test.eps}
\end{center}
\end{figure}
```

This includes the graphic stored in the file `test.eps`. The graphic is *first* rotated by 90 degrees and *then* scaled to the final width of 10 cm. The aspect ratio is 1.0 because no special height is specified.

For more information please refer to [8].

Bibliography

[1] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, Massachusetts, second edition, 1994, ISBN 0-201-52983-1.

[2] Donald E. Knuth. *The TeXbook,* Volume A of *Computers and Typesetting*, Addison-Wesley Publishing Company (1984), ISBN 0-201-13448-9.

[3] Michel Goossens, Frank Mittelbach and Alexander Samarin. *The LaTeX Companion.* Addison-Wesley, Reading, Massachusetts, 1994, ISBN 0-201-54199-8.

[4] Each LaTeX installation should provide a so-called *LaTeX Local Guide*, which explains the things which are special to the local system. It should be contained in file called `local.tex`. Unfortunately some lazy sysops do not provide such a document. In this case, go and ask you local LaTeX guru for help.

[5] LaTeX3 Project Team. *LaTeX 2ε for authors.* Comes with the LaTeX 2ε distribution as `usrguide.tex`.

[6] LaTeX3 Project Team. *LaTeX 2ε for Class and Package writers.* Comes with the LaTeX 2ε distribution as `clsguide.tex`.

[7] LaTeX3 Project Team. *LaTeX 2ε Font selection.* Comes with the LaTeX 2ε distribution as `fntguide.tex`.

[8] D. P. Carlisle. *Packages in the 'graphics' bundle.* Comes with the 'graphics' bundle as `grfguide.tex`, available form the same source your LaTeX distribution came from.