Chapter 2

Typesetting Text

After reading the previous chapter, you should know about the basic stuff of which a LaTeX 2$_\varepsilon$ document is made. In this chapter I will fill in the remaining structure you will need to know in order to produce real world material.

2.1 The Structure of Text and Language

By Hanspeter Schmid <hanspi@schmid-werren.ch>

The main point of writing a text, is to convey ideas, information, or knowledge to the reader. The reader will understand the text better if these ideas are well-structured, and will see and feel this structure much better if the typographical form reflects the logical and semantic structure of the content.

LaTeX is different from other typesetting systems in that you just have to tell it the logical and semantic structure of a text. It then derives the typographical form of the text according to the "rules" given in the document class file and in various style files.

The most important text unit in LaTeX (and in typography) is the paragraph. We call it "text unit" because a paragraph is the typographical form that should reflect one coherent thought, or one idea. You will learn in the following sections how to force line breaks with e.g. \\, and paragraph breaks with e.g. leaving an empty line in the source code. Therefore, if a new thought begins, a new paragraph should begin, and if not, only line breaks should be used. If in doubt about paragraph breaks, think about your text as a conveyor of ideas and thoughts. If you have a paragraph break, but the old thought continues, it should be removed. If some totally new line of thought occurs in the same paragraph, then it should be broken.

Most people completely underestimate the importance of well-placed paragraph breaks. Many people do not even know what the meaning of a paragraph break is, or, especially in LaTeX, introduce paragraph breaks without knowing it. The latter mistake is especially easy to make if equations are used in the text. Look at the following examples, and figure out why sometimes empty lines (paragraph breaks) are used before and after the equation, and sometimes not. (If you don't yet understand all commands well enough to understand these examples, please read this and the following chapter, and then read this section again.)

```
% Example 1
\ldots when Einstein introduced his formula
\begin{equation}
  e = m \cdot c^2 \; ,
\end{equation}
which is at the same time the most widely known
and the least well understood physical formula.


% Example 2
\ldots from which follows Kirchhoff's current law:
\begin{equation}
  \sum_{k=1}^{n} I_k = 0 \; .
\end{equation}

Kirchhoff's voltage law can be derived \ldots


% Example 3
\ldots which has several advantages.

\begin{equation}
  I_D = I_F - I_R
\end{equation}
is the core of a very different transistor model. \ldots
```

The next smaller text unit is a sentence. In English texts, there is a larger space after a period that ends a sentence than after one that ends an abbreviation. LaTeX tries to figure out which one you wanted to have. If LaTeX gets it wrong, you must tell it what you want. This is explained later in this chapter.

The structuring of text even extends to parts of sentences. Most languages have very complicated punctuation rules, but in many languages (including German and English), you will get almost every comma right if you remember what it represents: a short stop in the flow of language. If you are not sure about where to put a comma, read the sentence aloud and take a short breath at every comma. If this feels awkward at some place, delete that comma; if you feel the urge to breathe (or make a short stop) at some other place, insert a comma.

Finally, the paragraphs of a text should also be structured logically at a higher level, by putting them into chapters, sections, subsections, and so on. However, the typographical effect of writing e.g. `\section{The Structure of Text and Language}` is so obvious that it is almost self-evident how these high-level structures should be used.

2.2 Line Breaking and Page Breaking

2.2.1 Justified Paragraphs

Books are often typeset with each line having the same length. LaTeX inserts the necessary line breaks and spaces between words by optimizing 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 6.3.2 for more information.

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

`\\` or `\newline`

starts a new line without starting a new paragraph.

`\\*`

additionally prohibits a page break after the forced line break.

`\newpage`

starts a new page.

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

suggest places where a break may (or may not) happen. They enable the author to influence their actions with the optional argument n, which can be set to a number between zero and 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. Do not confuse these "break" commands with the "new" commands. Even when you give a "break" command, LaTeX still tries to even out the right border of the line and the total length of the page, as described in the next section; this can lead to unpleasant gaps in your text. If you really want to start a "new line" or a "new page", then use the corresponding command. Guess their names!

LaTeX always tries to produce the best line breaks possible. If it cannot find a way to break the lines in a manner that meets its high standards, it 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.[1] Instruct LaTeX to lower its standards a little by giving the `\sloppy` command. It 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 such cases the result doesn't look very good. The command `\fussy` brings LaTeX back to its default behaviour.

2.2.2 Hyphenation

LaTeX hyphenates words whenever necessary. If the hyphenation algorithm does not find the correct hyphenation points, 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 "`-`". The argument of the command should only contain words built from normal letters, or rather signs that are considered to be normal letters by LaTeX. The hyphenation hints are stored for the language that is active when the hyphenation command occurs. This means that if you place a hyphenation command into the preamble of your document it will influence the English language hyphenation. If you place the command after the `\begin{document}` and you are using some package for national language support like polyglossia, then the hyphenation hints will be active in the language activated through polyglossia.

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 special 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 (e.g. accented

[1]Although LaTeX gives you a warning when that happens (`Overfull \hbox`) and displays the offending line, such lines are not always easy to find. If you use the option `draft` in the `\documentclass` command, these lines will be marked with a thick black line on the right margin.

characters), because LaTeX does not automatically hyphenate words containing special characters.

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

I think this is: supercalifragilisticexpiali-docious

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

`\mbox{`*text*`}`

It causes its argument to be kept together under all circumstances.

```
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.
```

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

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

`\fbox` is similar to `\mbox`, but in addition there will be a visible box drawn around the content.

2.3 Ready-Made Strings

In some of the examples on the previous pages, you have seen some very simple LaTeX commands for typesetting special text strings:

Command	Example	Description
`\today`	March 9, 2021	Current date
`\TeX`	TeX	Your favorite typesetter
`\LaTeX`	LaTeX	The Name of the Game
`\LaTeXe`	LaTeX 2_ε	The current incarnation

2.4 Special Characters and Symbols

2.4.1 Quotation Marks

You should *not* use the `"` for quotation marks as you would on a typewriter. In publishing there are special opening and closing quotation marks. In LaTeX, use two `` ` `` (grave accent) for opening quotation marks and two `'` (vertical quote) for closing quotation marks. For single quotes you use just one of each.

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

“Please press the ‘x’ key.”

Yes I know the rendering is not ideal, it's really a back-tick or grave accent (`) for opening quotes and vertical quote (') for closing, despite what the font chosen might suggest.

2.4.2 Dashes and Hyphens

LaTeX knows four kinds of dashes. Access three of them with different number of consecutive dashes. The fourth sign is actually not a dash at all—it is the mathematical minus sign:

```
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.

2.4.3 Tilde ($\sim$)

A character often seen in web addresses is the tilde. To generate this in LaTeX use `\~{}` but the result (˜) is not really what you want. Try this instead:

```
http://www.rich.edu/\~{}bush \\
http://www.clever.edu/$\sim$demo
```

http://www.rich.edu/˜bush
http://www.clever.edu/$\sim$demo

2.4.4 Slash (/)

In order to typeset a slash between two words, one can simply type e.g. `read/write`, but this makes LaTeX treat the two words as one. Hyphenation is disabled for these two words, so there may be 'overfull' errors. To overcome this, use `\slash`. For example type '`read\slash write`' which allows hyphenation. But normal '`/`' character may be still used for ratios or units, e.g. `5 MB/s`.

2.4.5 Degree Symbol ($\circ$)

Printing the degree symbol in pure LaTeX.

```
It's $-30\,^{\circ}\mathrm{C}$.
I will soon start to
super-conduct.
```

It's $-30\,^{\circ}\mathrm{C}$. I will soon start to super-conduct.

The textcomp package makes the degree symbol also available as `\textdegree` or in combination with the C by using the `\textcelsius`.

```
30 \textcelsius{} is
86 \textdegree{}F.
```

30 ℃ is 86 °F.

2.4.6 The Euro Currency Symbol (€)

When writing about money these days, you need the Euro symbol. Many current fonts contain a Euro symbol. After loading the textcomp package in the preamble of your document

```
\usepackage{textcomp}
```

use the command

```
\texteuro
```

to access it.

If your font does not provide its own Euro symbol or if you do not like the font's Euro symbol, you have two more choices:

First the eurosym package. It provides the official Euro symbol:

```
\usepackage[official]{eurosym}
```

If you prefer a Euro symbol that matches your font, use the option `gen` in place of the `official` option.

Table 2.1: A bag full of Euro symbols

LM+textcomp	`\texteuro`	€	€	€
eurosym	`\euro`	€	€	€
[gen]eurosym	`\euro`	€	€	€

2.4.7 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, entering 'ellipsis' by just typing three dots would produce the wrong result. Instead, there is a special command for these dots. It is called

`\ldots` (low dots)

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

Not like this ... but like this:
New York, Tokyo, Budapest, …

2.4.8 Ligatures

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

ﬀ ﬁ ﬂ ﬃ… instead of ff fi fl ffi …

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

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

Not shelﬀul
but shelfful

2.4.9 Accents and Special Characters

LaTeX supports the use of accents and special characters from many languages. Table 2.2 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, its dots have to be removed. This is accomplished by typing `\i` and `\j`.

```
H\^otel, na\"\i ve, \'el\`eve,\\
sm\o rrebr\o d, !`Se\~norita!,\\
Sch\"onbrunner Schlo\ss{}
Stra\ss e
```

Hôtel, naïve, élève,
smørrebrød, ¡Señorita!,
Schönbrunner Schloß Straße

2.5 International Language Support

By Axel Kielhorn <A.Kielhorn@web.de>

When you write documents in languages other than English, there are three areas where LaTeX has to be configured appropriately:

1. All automatically generated text strings[2] have to be adapted to the new language.
2. LaTeX needs to know the hyphenation rules for the current language.
3. Language specific typographic rules. In French for example, there is a mandatory space before each colon character (:).

Also entering text in your language of choice might be a bit cumbersome using all the commands from figure 2.2. To overcome this problem, until recently you had to delve deep into the abyss of language specific encodings both for input as well as fonts. These days, with modern TeX engines speaking UTF-8 natively, these problems have relaxed considerably.

The package polyglossia[18] is a replacement for venerable babel package. It takes care of the hyphenation patterns and automatically generated text strings in your documents.

The package fontspec[20] handles font loading for XeLaTeX and LuaTeX. The default font is Latin Modern Roman.

2.5.1 Polyglossia Usage

Depending on the TeX engine you use slightly different commands are necessary in the preamble of your document to properly enable multilingual processing. Figure 2.1 on page 24 shows a sample preamble that takes care of all the necessary settings.

So far there has been no advantage to using a Unicode TeX engine. This changes when we leave the Latin script and move to a more interesting language like Greek or Russian. With a Unicode based system, you can

[2]Table of Contents, List of Figures, ...

Table 2.2: Accents and Special Characters.

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

simply[3] enter the native characters in your editor and TEX will understand them.

Writing in different languages is easy, just specify the languages in the preamble. This example uses the csquotes package which generates the right kind of quotes according to the language you are writing in. Note that it needs to be loaded *before* loading the language support.

```
\usepackage[autostyle=true]{csquotes}
\setdefaultlanguage{english}
\setotherlanguage{german}
```

To write a paragraph in German, you can use the German environment:

```
English text.
\begin{german}
Deutscher \enquote{Text}.
\end{german}
More English \enquote{text}.
```

English text. Deutscher „Text“. More English “text”.

If you just need a word in a foreign language you can use the `\text`*language* command:

```
Did you know that
\textgerman{Gesundheit} is
actually a German word.
```

Did you know that Gesundheit is actually a German word.

This may look unnecessary since the only advantage is a correct hyphenation, but when the second language is a little bit more exotic it will be worth the effort.

Sometimes the font used in the main document does not contain glyphs that are required in the second language. Latin Modern for example does

[3] For small values of simple.

```
\usepackage{iftex}
\ifXeTeX
   \usepackage{fontspec}
\else
   \usepackage{luatextra}
\fi
\defaultfontfeatures{Ligatures=TeX}
\usepackage{polyglossia}
```

Figure 2.1: All in one preamble that takes care of LuaLATEX and XƎLATEX

not contain Cyrillic letters. The solution is to define a font that will be used for that language. Whenever a new language is activated, polyglossia will first check whether a font has been defined for that language. If you are happy with the computer modern font, you may want to try the "Computer Modern Unicode" font by adding the following commands to the preamble of your document.

For LuaLaTeX it is pretty simple

```
\setmainfont{CMU Serif}
\setsansfont{CMU Sans Serif}
\setmonofont{CMU Typewriter Text}
```

For XeLaTeX you have to be a bit more explicit:

```
\setmainfont{cmun}[
   Extension=.otf,UprightFont=*rm,ItalicFont=*ti,
   BoldFont=*bx,BoldItalicFont=*bi,
 ]
 \setsansfont{cmun}[
   Extension=.otf,UprightFont=*ss,ItalicFont=*si,
   BoldFont=*sx,BoldItalicFont=*so,
 ]
 \setmonofont{cmun}[
   Extension=.otf,UprightFont=*btl,ItalicFont=*bto,
   BoldFont=*tb,BoldItalicFont=*tx,
 ]
```

With the appropriate fonts loaded, you can now write:

```
\textrussian{Правда} is
a russian newspaper.
\textgreek{ἀλήθεια} is truth
or disclosure in philosophy
```

Правда is a russian newspaper. ἀλήθεια is truth or disclosure in philosophy

The package xgreek[21] offers support for writing either ancient or modern (monotonic or polytonic) greek.

Right to Left (RTL) languages.

Some languages are written left to right, others are written right to left(RTL). polyglossia needs the bidi[22] package[4] in order to support RTL languages. The bidi package should be the last package you load, even after hyperref which is usually the last package. (Since polyglossia loads bidi this means that polyglossia should be the last package loaded.)

[4]bidi does not support LuaTeX.

The package xepersian[23] offers support for the Persian language. It supplies Persian LaTeX-commands that allows you to enter commands like `\section` in Persian, which makes this really attractive to native speakers. xepersian is the only package that supports kashida with XeLaTeX. A package for Syriac which uses a similar algorithm is under development.

The IranNastaliq font provided by the SCICT[5] is available at their website http://www.scict.ir/Portal/Home/Default.aspx.

The arabxetex[19] package supports several languages with an Arabic script:

- arab (Arabic)
- persian
- urdu
- sindhi
- pashto
- ottoman (turk)
- kurdish
- kashmiri
- malay (jawi)
- uighur

It offers a font mapping that enables XeLaTeX to process input using the ArabTeX ASCII transcription.

Fonts that support several Arabic laguages are offered by the IRMUG[6] at http://wiki.irmug.org/index.php/X_Series_2.

There is no package available for Hebrew because none is needed. The Hebrew support in polyglossia should be sufficient. But you do need a suitable font with real Unicode Hebrew. SBL Hebrew is free for non-commercial use and available at http://www.sbl-site.org/educational/biblicalfonts.aspx. Another font available under the Open Font License is Ezra SIL, available at http://www.sil.org/computing/catalog/show_software.asp?id=76.

Remember to select the correct script:

```
\newfontfamily\hebrewfont[Script=Hebrew]{SBL Hebrew}
\newfontfamily\hebrewfont[Script=Hebrew]{Ezra SIL}
```

[5]Supreme Council of Information and Communication Technology
[6]Iranian Mac User Group

Chinese, Japanese and Korean (CJK)

The package xeCJK[24] takes care of font selection and punctuation for these languages.

2.6 The Space Between Words

To get a straight right margin in the output, LaTeX inserts varying amounts of space between the words. It inserts slightly more space at the end of a sentence, 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 uppercase letters normally occur in abbreviations.

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

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

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

The additional space after periods can be disabled with the command

```
\frenchspacing
```

which tells LaTeX *not* to insert more space after a period than after an ordinary character. This is very common in non-English languages, except bibliographies. If you use `\frenchspacing`, the command `\@` is not necessary.

2.7 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 that take the section title as their argument. It is up to you to use them in the correct order.

The following sectioning commands are available for the `article` class:

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

If you want to split your document into parts without influencing the section or chapter numbering use

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

When you work with the `report` or `book` class, an additional top-level sectioning command becomes available

```
\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 numbering to letters.[7]

LaTeX creates a table of contents by taking the section headings and page numbers from the last compile cycle of the document. The command

```
\tableofcontents
```

expands to a table of contents at the place it is issued. A new document has to be compiled ("LaTeXed") twice to get a correct table of contents. Sometimes 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. This generates section headings that do not show up in the table of contents and are not numbered. The command `\section{Help}`, for example, would become `\section*{Help}`.

[7]For the article style it changes the section numbering.

Normally the section headings show up in the table of contents exactly as they are 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 then be specified as an optional argument in front of the actual heading.

```
\chapter[Title for the table of contents]{A long
    and especially boring title, shown in the text}
```

The title of the whole document is generated by issuing a

```
\maketitle
```

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

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

before calling `\maketitle`. In the argument to `\author`, you can supply several names separated by `\and` commands.

An example of some of the commands mentioned above can be found in Figure 1.2 on page 8.

Apart from the sectioning commands explained above, LaTeX 2$_\varepsilon$ introduced three additional commands for use with the `book` class. They are useful for dividing your publication. The commands alter chapter headings and page numbering to work as you would expect in a book:

`\frontmatter` should be the very first command after the start of the document body (`\begin{document}`). It will switch page numbering to Roman numerals and sections will be non-enumerated as if you were using the starred sectioning commands (eg `\chapter*{Preface}`) but the sections will still show up in the table of contents.

`\mainmatter` comes right before the first chapter of the book. It turns on Arabic page numbering and restarts the page counter.

`\appendix` marks the start of additional material in your book. After this command chapters will be numbered with letters.

`\backmatter` should be inserted before the very last items in your book, such as the bibliography and the index. In the standard document classes, this has no visual effect.

2.8 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 after which the corresponding `\label` command was issued. `\pageref` prints the page number of the page where the `\label` command occurred.[8] As with section titles and page numbers for the table of contents, the numbers from the previous compile cycle are used.

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

A reference to this subsection looks like: “see section 2.8 on page 30.”

2.9 Footnotes

With the command

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

a footnote is printed at the foot of the current page. Footnotes should always be put[9] after the word or sentence they refer to. Footnotes referring to a sentence or part of it should therefore be put after the comma or period.[10]

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

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

[a]This is a footnote.

[8]Note that these commands are not aware of what they refer to. `\label` just saves the last automatically generated number.

[9]“put” is one of the most common English words.

[10]Note that footnotes distract the reader from the main body of your document. After all, everybody reads the footnotes—we are a curious species, so why not just integrate everything you want to say into the body of the document?[11]

[11]A guidepost doesn’t necessarily go where it’s pointing to :-).

2.10 Emphasized Words

If a text is typed using a typewriter, important words are `emphasized by underlining them.`

```
\underline{text}
```

In printed books, however, words are emphasized by typesetting them in an *italic* font. As an author you shouldn't care either way. The important bit is, to tell LaTeX that a particular bit of text is important and should be emphasized. Hence the command

```
\emph{text}
```

to emphasize text. What the command actually does with its argument depends on the context:

```
\emph{If you use
  emphasizing inside a piece
  of emphasized text, then
  \LaTeX{} uses the
  \emph{normal} font for
  emphasizing.}
```

If you use emphasizing inside a piece of emphasized text, then LaTeX uses the nor-mal *font for emphasizing.*

If you want control over font and font size, section 6.2 on page 107 might provide some inspiration.

2.11 Environments

```
\begin{environment}   text   \end{environment}
```

Where *environment* is the name of the environment. Environments can be nested within each other as long as the correct nesting order is maintained.

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

In the following sections all important environments are explained.

2.11.1 Itemize, Enumerate, and Description

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

```
\flushleft
\begin{enumerate}
\item You can nest the list
environments to your taste:
\begin{itemize}
\item But it might start to
look silly.
\item[-] With a dash.
\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}
```

1. You can nest the list environments to your taste:
   - But it might start to look silly.
   - With a dash.
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.

2.11.2 Flushleft, Flushright, and Center

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

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

This text is
left-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}
```

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

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

At the centre
of the earth

2.11.3 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:
\begin{quote}
On average, no line should
be longer than 66 characters.
\end{quote}
This is why \LaTeX{} pages have
such large borders by default
and also why multicolumn print
is used in newspapers.
```

A typographical rule of thumb for the line length is:

> On average, no line should be longer than 66 characters.

This is why LaTeX pages have such large borders by default and also why multicolumn print is 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, because it indents the first line of each paragraph. 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 an empty line after each verse.

```
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}
```

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.

2.11.4 Abstract

In scientific publications it is customary to start with an abstract which gives the reader a quick overview of what to expect. LaTeX provides the `abstract` environment for this purpose. Normally `abstract` is used in documents typeset with the article document class.

```
\begin{abstract}
The abstract abstract.
\end{abstract}
```

The abstract abstract.

2.11.5 Printing Verbatim

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

Within a paragraph, similar behavior can be accessed with

```
\verb+text+
```

The `+` is just an example of a delimiter character. Use any character except letters, `*` or space. Many LaTeX examples in this booklet are typeset with this command.

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

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

```
The \ldots command ...


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

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

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

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

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

```
like␣␣␣this␣:-)␣
```

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

2.11.6 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}[pos]{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 line breaks, and `|` for a vertical line.

If the text in a column is too wide for the page, LaTeX won't automatically wrap it. Using `p{`*`width`*`}` you can define a special type of column which will wrap-around the text as in a normal paragraph.

The *pos* argument specifies the vertical position of the table relative to the baseline of the surrounding text. Use one of the letters `t`, `b` and `c` to specify table alignment at the top, bottom or centre.

Within a `tabular` environment, `&` jumps to the next column, `\\` starts a new line and `\hline` inserts a horizontal line. Add partial lines by using `\cline{`*i*`-`*j*`}`, where *i* and *j* are the column numbers the line should extend over.

```
\begin{tabular}{|r|l|}
\hline
7C0 & hexadecimal \\
3700 & octal \\ \cline{2-2}
11111000000 & binary \\
\hline \hline
1984 & decimal \\
\hline
\end{tabular}
```

7C0	hexadecimal
3700	octal
11111000000	binary
1984	decimal

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

Welcome to Boxy's paragraph.
We sincerely hope you'll all en-
joy the show.

The column separator can be specified with the `@{...}` construct. This command kills the inter-column space and replaces it with whatever is between the curly braces. One common use for this command is explained below in the decimal alignment problem. Another possible application is to suppress leading space in a table with `@{}`.

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

no leading space

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

leading space left and right

Since there is no built-in way to align numeric columns to a decimal point,[12] 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 inter-column 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.

```
\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}
```

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

```
\begin{tabular}{|c|c|}
\hline
\multicolumn{2}{|c|}{Ene} \\
\hline
Mene & Muh! \\
\hline
\end{tabular}
```

Ene	
Mene	Muh!

Material typeset with the tabular environment always stays together on one page. If you want to typeset long tables, you might want to use the longtable environments.

Sometimes the default LaTeX tables do feel a bit cramped. So you may want to give them a bit more breathing space by setting a higher `\arraystretch` and `\tabcolsep` value.

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

```
\begin{tabular}{|l|}
\hline
These lines\\\hline
are tight\\\hline
\end{tabular}

{\renewcommand{\arraystretch}{1.5}
\renewcommand{\tabcolsep}{0.2cm}
\begin{tabular}{|l|}
\hline
less cramped\\\hline
table layout\\\hline
\end{tabular}}
```

These lines
are tight

less cramped
table layout

If you just want to grow the height of a single row in your table add an invisible vertical bar[13]. Use a zero width `\rule` to implement this trick.

```
\begin{tabular}{|c|}
\hline
\rule{1pt}{4ex}Pitprop \ldots\\
\hline
\rule{0pt}{4ex}Strut\\
\hline
\end{tabular}
```

|Pitprop …
Strut

The `pt` and `ex` in the example above are TEX units. Read more on units in table 6.5 on page 114.

A number of extra commands, enhancing the tabular environment are available in the booktabs package. It makes the creation of professional looking tables with proper spacing quite a bit simpler.

2.12 Including Graphics and Images

As explained in the previous section LATEX provides the facilities to work with floating bodies, such as images or graphics, with the `figure` and `table` environments.

A good set of commands for inclusion of graphics into these floating bodies is provided in the graphicx package by D. P. Carlisle. It is part of a whole family of packages called the "graphics" bundle.[14]

Use the following step by step guide to include a picture into your document:

1. Export the picture from your graphics program in EPS, PDF, PNG or JPEG format.

[13]In professional typesetting, this is called a strut.
[14]CTAN://pkg/graphics

Table 2.3: Key Names for graphicx Package.

`width`	scale graphic to the specified width
`height`	scale graphic to the specified height
`angle`	rotate graphic counterclockwise
`scale`	scale graphic

```
\includegraphics[angle=90,width=\textwidth]{test.png}
```

Figure 2.2: Example code for including `test.png` into a document.

2. If you exported your graphics as an EPS vector graphics, you have to convert it to PDF format prior to using it. There is a `epstopdf` command line tool that helps with this task. Note that it may be sensible to export EPS eventhough your software can export PDF too, as PDFs often are full page and will thus get very small when imported into a document. EPS on the other hand come with a bounding box showing the extent of the actual graphics.

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

   ```
   \usepackage{graphicx}
   ```

4. Use the command

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

   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 2.3 lists the most important keys.

The example code in figure 2.2 on page 38 may help to clarify things. It includes the graphic stored in the file `test.png`. The graphic is *first* rotated by an angle of 90 degrees and *then* scaled to the final width of 0.5 times the width of a standard paragraph. The aspect ratio is 1.0, because no special height is specified. The width and height parameters can also be specified in absolute dimensions. Refer to Table 6.5 on page 114 for more information. If you want to know more about this topic, make sure to read [9].

2.13 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 that 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:

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}[...]`

called the *placement specifier*. This parameter is used to tell LaTeX about the locations to which the float is allowed to be moved. A *placement specifier* is constructed by building a string of *float-placing permissions*. See Table 2.4.

For example, a table could be started with the following line

```
   \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

Table 2.4: 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 otherwise stop this float from being placed.

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

all this 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* queue or the *tables* queue.[15] 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 it had just occurred in the text: LaTeX tries again to place it according to its 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 that cannot be placed pushes all 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.

While it is possible to give LaTeX single-location placement specifiers, this causes problems. If the float does not fit in the location specified it becomes stuck, blocking subsequent floats. In particular, you should never, ever use the [h] option—it is so bad that in more recent versions of LaTeX, it is automatically replaced by [ht].

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

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

command to 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. These lists will display the whole caption, so if you tend to use long captions you must have a shorter version of the caption for the lists. This is accomplished by entering the short version in brackets after the `\caption` command.

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

[15]These are FIFO—'first in first out'—queues!

Use `\label` and `\ref` to create a reference to a float within your text. Note that the `\label` command must come *after* the `\caption` command since you want it to reference the number of the caption.

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}[!hbtp]
\includegraphics[angle=90,width=\textwidth]{white-box.pdf}
\caption{White Box by Peter Markus Paulian.\label{white}}
\end{figure}
```

In the example above, LaTeX will try *really hard* (`!`) to place the figure right *here* (`h`).[16] 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 whether 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` or even the `\cleardoublepage`

command. It orders LaTeX to immediately place all floats remaining in the queues and then start a new page. `\cleardoublepage` even goes to a new right-hand page.

[16]assuming the figure queue is empty.