
Text

In Chapter 1, we briefly discussed how to type text in a document. Now we take up this topic more fully.

This chapter starts with Section 2.1, a discussion of words, sentences, and paragraphs. In Section 2.2, we are introduced to commands and environments.

A document can contain text symbols that cannot be found on your keyboard. In Section 2.3, we show how to get these symbols in our typeset documents by using commands.

Some other characters are defined by \LaTeX as command characters. For example, the % character plays a special role in the source document. In Section 2.4.1, you will see how % is used to comment out lines. In Section 2.4.2, we introduce the command for footnotes.

In Section 2.5, we discuss the commands (and environments) for changing font shapes and sizes. In Section 2.6, you learn about lines, paragraphs, and pages. The judicious use of horizontal and vertical spacing is an important part of document formatting, and also the topic of Section 2.7. In Section 2.8, you learn how to typeset text in a “box”, which behaves as if it were a single large character.

2.1 Words, sentences, and paragraphs

Text consists of words, sentences, and paragraphs. In text, *words* are separated by one or more spaces, which can include a single end-of-line character (see the rule, **Spacing in text**), or by parentheses and punctuation marks. A group of words terminated by a period, exclamation point, or question mark forms a *sentence*. A group of sentences terminated by one or more blank lines constitutes a *paragraph*.

2.1.1 Spacing rules

Here are the most important L^AT_EX rules about spaces in text in the source file.

Practical Rule ■ Spacing in text

1. Two or more spaces in text are the same as one.
 2. A tab or end-of-line character is the same as a space.
 3. A blank line, that is, two end-of-line characters separated only by spaces and tabs, indicates the end of a paragraph. The `\par` command does the same.
 4. Spaces at the beginning of a line are ignored.
-

Rules 1 and 2 make cutting and pasting text less error-prone. However,

```
the number of   spaces separating words,
as long
```

and

```
the number of   spaces separating words
, as long
```

produce different results:

```
┌
the number of spaces separating words, as long
the number of spaces separating words , as long
└
```

Notice the space between “words” and the comma in the second line. That space was produced by the end-of-line character in accordance with Rule 2.



Practical Tip 15. It is very important to maintain the readability of your source file. L^AT_EX does not care about the number of spaces or line length, but you, your coauthor, or your editor might.

2.1.2 Periods

L^AT_EX places a certain size space between words—the *interword space*—and a somewhat larger space between sentences—the *intersentence space*. To know which space to use, L^AT_EX must decide whether or not a period indicates the end of a sentence.

Practical Rule 1 ■ Period

To L^AT_EX, a period after a capital letter, for instance, A. or caT., signifies an abbreviation or an initial. Generally, every other period signifies the end of a sentence.

This rule works most of the time. When it fails—for instance, twice with e.g.—you need to specify the type of space you want, using the following two rules.

Practical Rule 2 ■ Period

If an abbreviation does not end with a capital letter, for instance, etc., and it is not the last word in the sentence, then follow the period by an interword space (`_`) or a tie (`~`), if appropriate.

Recall that `_` provides an interword space.

The result was first published, in a first approximation,
in the Combin.\ Journal. The result was first published,
in a first approximation, in the Combin. Journal.

prints as

```
[
The result was first published, in a first approximation, in the Combin. Journal.
The result was first published, in a first approximation, in the Combin. Journal.
]
```

Notice that `Combin.` in the first line is followed by a regular interword space. The incorrect intersentence space following `Combin.` in the second line is a little wider.



Practical Tip 16. The `thebibliography` environment handles periods properly. You do not have to mark periods for abbreviations (in the form `._`) in the name of a journal, so

`Acta Math. Acad. Sci. Hungar.`

is correct.

The next rule contradicts Rules 1 and 2; consider it an exception.

Practical Rule 3 ■ Period

If a capital letter is followed by a period and is at the end of a sentence, precede the period with `\@`.

For example,

```
(1) follows from condition~H\@. We can proceed\\
(1) follows from condition~H. We can proceed
```

prints:

```
┌
(1) follows from condition H. We can proceed
(1) follows from condition H. We can proceed
└
```

Notice that there is not enough space after H. in the second line.

Most typographers agree on the following rule:

Practical Rule 4 ■ Period

Add no space or a thin space (`\,`) within strings of initials and be consistent.

So W.H. Lampstone with no space or W.H. Lampstone with thin space is preferred over W. H. Lampstone.

2.2 Commanding \LaTeX

How do you tell \LaTeX to do something special for you, such as starting a new line, changing emphasis, or displaying the next theorem? You use *commands* and special pairs of commands called *environments*, both briefly introduced in Section 1.12.

Many commands have *arguments*, which are usually fairly brief. Environments have *contents*, the text between the `\begin` and `\end` commands. The contents of an environment can be several paragraphs long.

2.2.1 Commands and environments

The `\emph{Careful!}` *command* instructs \LaTeX to emphasize its argument: *Careful!* The `\&` command has no argument. It instructs \LaTeX to typeset `&`; see Section 1.2.

The *center environment* instructs \LaTeX to center the contents, the text between the two commands `\begin{center}` and `\end{center}`. The body of the document (see Section 1.9) is the contents of the *document environment* and the abstract is the contents of the *abstract environment*.

Practical Rule ■ Environments

An environment starts with the command `\begin{name}` and ends with `\end{name}`. Between these two lines is the *contents* of the environment, affected by the definition of the environment.

Practical Rule ■ Commands

A L^AT_EX command starts with a backslash, \, and is followed by the *command name*. The *name* of a command is either a *single non-alphabetic character* other than a tab or end-of-line character or a *string of letters*, that is, one or more letters.

So # and ' are valid command names. The corresponding commands \# and \' are discussed in Sections 1.2. More valid command names: input and date. However, input3, in#ut, and in_ut are not valid names because 3, #, and _ should not occur in a multicharacter command name. Note that _ is a command name; the command _ produces a blank.

L^AT_EX has a few commands, for instance, \$ (see Section 1.5) that do not follow this naming scheme, that is, they are not of the form \name.

Practical Rule ■ Command termination

L^AT_EX finds the end of a command name as follows:

- If the first character of the name is not a letter, the name is the first character.
- If the first character of the name is a letter, the command name is terminated by the first nonletter.

If the command name is a string of letters, and is terminated by a space, then L^AT_EX discards all spaces following the command name.

While emph3 is an invalid name, \emph3 is not an incorrect command. It is the \emph command followed by the character 3, which is either part of the text following the command or the argument of the command.

L^AT_EX also allows some command names to be modified with *. Such commands are referred to as **-ed commands*. Many commands have *-ed variants. \hspace* is an often-used *-ed command; see Section 2.7.1.

Practical Rule ■ Command and environment names

Command and environment names are *case sensitive*. \ShowLabels is not the same as \showlabels.

Practical Rule ■ Arguments

Arguments are enclosed in braces, { }.

Optional arguments are enclosed in brackets, [].

Commands can have *arguments*, typed in braces immediately after the command. The argument(s) are used in processing the command. Accents provide very simple

examples. For instance, `\'o`—which produces \acute{o} —consists of the command `\'` and the argument `o`; see Sections 1.3 and 2.3.1. In `\emph{together}`, the command is `\emph` and the argument is `together`.

Some environments also have arguments. For example, the `alignat` environment (see Section 5.3.1) is delimited by the commands

```
\begin{alignat}{2} and \end{alignat}
```

The argument, 2, is the number of columns—it could be any number 1, 2, ... A command or environment can have more than one argument. The `\frac` command has two; `\frac{1}{2}` typesets as $\frac{1}{2}$.

Some commands and environments have one or more *optional arguments*, that is, arguments that may or may not be present. The `\sqrt` command (see Section 1.7) has an optional argument for specifying roots other than the square root. To get $\sqrt[3]{25}$, type `\sqrt[3]{25}`. The `\documentclass` command has an argument, the name of a document class, and an optional argument, a list of options (see Section 6.2), for instance,

```
\documentclass[12pt,draft]{amsart}
```



Practical Tip 17. If you get an error when using a command, check that:

1. The command is spelled correctly, including the use of uppercase and lowercase letters.
2. You have specified all required arguments in braces.
3. Any optional argument is in brackets, not braces or parentheses.
4. The command is properly terminated.
5. The package providing the command is loaded with the `\usepackage` command.

Most errors in the use of commands are caused by breaking the termination rule. We illustrate some of these errors with the `\today` command, which produces today's date. You have already seen this command in Section 1.3. The correct usage is `\today\` or `\today{}`. In the first case, `\today` was terminated by `\`, the command that produces an interword space. In the second case, it was terminated by the *empty group* `{}`.

If there is no space after the `\today` command, as in

```
\todayis\the\day
```

you get the error message

```
! Undefined control sequence.
```

```
1.6 \todayis
```

```
the day
```


\LaTeX thinks that `\todayis` is the command, and, of course, does not recognize it.

If you type one or more spaces after `\today`:

```
\today is the day
```

L^AT_EX interprets the two spaces as a single space by the first space rule (see page 32), and uses that one space to delimit `\today` from the text that follows it. So L^AT_EX produces

```
[
  March 19, 2014 is the day
]
```

 **Practical Tip 18.** If a command—or environment—can have an optional argument and none is given, and the text following the command starts with `[`, then type this as `{[}`.

This can happen, for instance, with the command `\item` (see page 54).

2.2.2 Scope

A command issued inside a pair of braces `{ }` has no effect beyond the right brace. You can have many braces:

```
{ ... { ... { ... } ... } ... }
```

The innermost pair containing a command is the *scope* of that command. The command has no effect outside its scope. We can illustrate this concept using the `\bfseries` command that switches the font to boldface:

```
{some text \bfseries bold text} no more bold
```

typesets as

```
[
  some text bold text no more bold
]
```

The commands `\begin{name}` and `\end{name}` bracketing an environment act also as a pair of braces and so delimit the scope. Also, `$`, `\`, `[`, and `\]` are special braces.

Practical Rule ■ Braces

1. Braces must be balanced: An opening brace (left brace) must have a matching closing brace (right brace), and a closing brace (right brace) must have a matching opening brace.
 2. Pairs of braces cannot overlap.
-

Violating the first brace rule generates warnings and error messages. If there is one more opening brace than closing brace, the document typesets, but you get a warning:

```
(\end occurred inside a group at level 1)
```

For two or more unmatched opening braces, you are warned that `\end` occurred inside a group at level 2, and so on.



Practical Tip 19. Do not disregard such warnings even if the document is already correctly typeset. At a later time, such errors can have strange consequences.

2.2.3 Types of commands

It can be useful at this point to note that commands can be of various types.

Some commands have arguments, and some do not. Some commands effect change only in their arguments, while some commands declare a change. For instance, `\textbf{This is bold}` typesets the phrase **This is bold** in bold type: **This is bold** and has no effect on the text following the argument of the command. On the other hand, the command `\bfseries` declares that the text that follows should be bold. This command has no argument. I call a command that declares change a *command declaration*. So `\bfseries` is a command declaration, while `\textbf` is not. As a rule, command declarations are commands without arguments.

Commands with arguments are called *long*—or commands with long arguments—if their argument(s) can contain a blank line or a `\par` command; otherwise they are *short*—or commands with short arguments. For example, `\textbf` is a short command. So are all the top matter commands discussed in Section 6.6.1.

Fragile commands

As a rule, \LaTeX reads a paragraph of the source file, typesets it, and then goes on to the next paragraph. Some information from the source file, however, is separately stored for later use.

Examples: the title of a document, which is reused as a running head (Section 6.6.1), table of contents (Sections 6.5.4), or footnote (Section 2.4.2).

These are *movable arguments*, and certain commands embedded in them must be protected from damage while being moved. \LaTeX commands that need such protection are called *fragile*. The inline math delimiter commands `\(` and `\)` are fragile, while `$` is not. In a movable argument, fragile commands must be protected with a `\protect` command. Thus `\(f(x^{\{2\}}) \)` is not appropriate in the title for a document, but

```
\protect \( f(x^{\{2\}} ) \protect \)
```

is. To be on the safe side, you should protect every command that might cause problems in a movable argument. A user-defined command, declared with

```
\DeclareRobustCommand
```

is not fragile; it needs no protection. This command is like the `\newcommand` (see Section 7.1) but defines a *robust* command.

2.3 Symbols not on the keyboard

A typeset document can contain symbols that cannot be typed. Some of these symbols can even be available on the keyboard but you are prohibited from using them. In this section, we discuss the commands that typeset some of these symbols in text.

Quotation marks To produce single and double quotes, as in

┌
└

‘subdirectly irreducible’ and “subdirectly irreducible”

type

‘subdirectly irreducible’ and ‘‘subdirectly irreducible’’

Here, ‘ is the left single quote and ’ is the right single quote.



Practical Tip 20. The double quote is obtained by typing the single quote key twice, and *not* by using the double quote key.

Dashes: hyphens A *hyphen*, -, is used to connect words:

┌
└

Mean-Value Theorem

This phrase is typed with a single dash:

Mean-Value Theorem

Dashes: en dashes An *en dash*, –, is typed as -- and is used for number ranges; for instance, the phrase see pages 23–45, is typed as

see pages~23--45

where ~ is a nonbreakable space (see Section 1.3), which is used to avoid having pages at the end of one line and 23–45 at the beginning of the next line.

Dashes: em dashes A long dash—called an *em dash*—is used to mark a change in thought or to add emphasis to a parenthetical clause, as in this sentence. The two em dashes in the last sentence are typed as follows:

A long dash---called an \emph{em dash}---is used

Note that there is no space before or after an en dash or em dash and en dash or em dash in a formula except in the argument of a \text command.

Ties or nonbreakable spaces A *tie* or *nonbreakable space* is an interword space that cannot be broken across lines. For instance, when referencing P. Neukomm in a document, you do not want the initial P. at the end of a line and the surname Neukomm at the beginning of the next line. To prevent this, you should type P.~Neukomm.

The following examples show some typical uses:

Theorem~\ref{T:main} in Section~\ref{S:intro}

the lattice~\$L\$.

Sections~\ref{S:modular} and~\ref{S:distributive}

In~\$L\$, we find

Ellipses The text ellipsis, ..., is produced using the `\dots` command. Typing three periods produces ... (notice that the spacing is wrong).

Ligatures Certain groups of characters, when typeset, are joined together—such compound characters are called *ligatures*. There are five ligatures that \LaTeX typesets automatically: ff, fi, fl, ffi, and ffl.

If you want to prevent \LaTeX from forming a ligature, separate the characters with an empty group `{ }` (officially, with `\textcompwordmark`). Compare iff with iff, typed as iff and `if{}f`.

2.3.1 Accents and symbols in text

\LaTeX provides 15 text accents. Type the command for the accent (`\` and a character), followed by the letter (in braces) on which you want the accent placed; see Section A.2.

For example, to get Grätzer György, type `Gr\"atzer Gy\"orgy` and to get Ö type `\"O`.

To place an accent on top of an i or a j, you must use the *dotless* version of i and j. These are obtained by the commands `\i` and `\j`: `\'i` typesets as í and `\v{j}` typesets as ĵ.

Sections A.1 and A.2 list European characters and text accents available in \LaTeX . Section A.4 adds some extra text symbols.

2.3.2 Logos and useful numbers

`\TeX` produces \TeX and `\LaTeX` produces \LaTeX .

Remember to type `\TeX_\square` or `\TeX{ }` if you need a space after \TeX (similarly for the others). A better way to handle this problem is discussed on page 113.

\LaTeX also stores some useful numbers:

- `\day` is the day of the month
- `\month` is the month of the year
- `\year` is the current year

You can include these numbers in your document by using the `\the` command:

Year: `\the\year`; month: `\the\month`; day: `\the\day`

produces a result such as

```
┌
Year: 2014; month: 1; day: 11
└
```

Of more interest is the `\today` command, which produces today's date in the form: January 11, 2014. It is often used as the argument of the `\date` command; see Section 1.3.

2.3.3 Hyphenation

In Section 1.4 we discussed optional hyphens.

Practical Rule ■ Hyphenation specifications

In the preamble, list the words that often need help in a command:

```
\hyphenation{set-up as-so-ciate}
```

All occurrences of the listed words will be hyphenated as specified.

Note that in the `\hyphenation` command the hyphens are designated by `-` and not by `\-`, and that the words are separated by spaces, not by commas.

You must use optional hyphens for words with accented characters, as in

```
Gr\{"a}t\~zer
```

Practical Rule ■ Preventing hyphenation

To *prevent* hyphenation of a word, put it in the argument of a `\text` command or place it unhyphenated in a `\hyphenation` command.

For example, type

```
\text{database}
```

if you do not want this instance of `database` hyphenated, or type

```
\hyphenation{database}
```

if you do not want \LaTeX to hyphenate any occurrence of the word anywhere after this command in your document. Typing `data\~base` overrides the general prohibition for this one instance.

2.4 Comments and footnotes

Various parts of your source file do not get typeset like the rest. The two primary examples are comments that do not get typeset at all and footnotes that get typeset at the bottom of the page.

2.4.1 Comments

The % symbol tells L^AT_EX to ignore the rest of the line. For instance, making a note to look up the proper reference:

therefore, a reference to Theorem~15 % check this!

The % symbol has many uses. For instance,

```
\documentclass[twocolumn,twoside]{amsart}
```

can be typed with explanations, as

```
\documentclass[twocolumn,% option for two-column pages
twoside,% format for two-sided printing
]{amsart}
```

so you can easily comment out some options at a later time.



Practical Tip 21. Some command arguments do not allow any spaces. If you want to break a line within an argument list, you can terminate the line with a %, as shown in the previous example.



Practical Tip 22. The 25% rule

If you want a % sign in text, make sure you type it as \%. Otherwise, % comments out the rest of the line. L^AT_EX does not produce a warning.

Using % to comment out large blocks of text can be tedious even with block comment. The `verbatim` package includes the `comment` environment:

```
\begin{comment}
...the commented out text...
\end{comment}
```

Practical Rule ■ The `comment` environment

1. `\end{comment}` must be at the beginning of a line by itself.
 2. There can be no comment within a comment.
-

The `comment` environment can be very useful in locating errors.



Practical Tip 23. Suppose you have unbalanced braces in your source file (see Section 2.2.2). Working with a *copy* of your source file, comment out the first half at a safe point (not within an environment!) and typeset. If you still get the same error message, the error is in the second half. If there is no error message, the error is in the first half. Comment out the half that has no error.

Now comment out half of the remaining text and typeset again. Check to see whether the error appears in the first half of the remaining text or the second. Continue applying this method until you narrow down the error to a paragraph that you can inspect visually.

Since the `comment` environment requires the `verbatim` package, you must include the line

```
\usepackage{verbatim}
```

in the preamble of the source file; see Section 1.9.

2.4.2 Footnotes

A footnote is typed as the argument of a `\footnote` command. To illustrate the use of footnotes, I have placed one here.¹ This footnote is typed as

```
\footnote{Footnotes are easy to place.}
```

2.5 Changing font characteristics

Although a document class and its options determine how \LaTeX typesets characters, there are occasions when you want control over the shape or size of the font used.

2.5.1 Basic font characteristics

You do not have to be a typesetting expert to recognize the following basic font attributes:

Shape Normal text is typeset:

<i>upright</i> (or <i>roman</i>)	as this text
<i>slanted</i>	as this text
<i>italic</i>	as this text
<i>small caps</i>	AS THIS TEXT

Monospaced and proportional Typewriters used *monospaced* fonts, that is, fonts all of whose characters are of the same width. Most text editors display text using a monospaced font. \LaTeX calls monospaced fonts *typewriter style*. In this book, such a font is used to represent user input and \LaTeX 's response, such as “`typewriter style text`”. Whereas, normal text is typeset in a *proportional* font, such as “proportional text with ii and mm”, in which i is narrow and m is wide.

Serifs A *serif* is a small horizontal (sometimes vertical) stroke used to finish off a vertical stroke of a letter, as on the top and bottom of the letter M. \LaTeX 's standard serif font is Computer Modern roman, such as “serif text”. Fonts without serifs are called *sans serif*, such as “sans serif text”. Sans serif fonts are often used for titles or for special emphasis.

Series: weight and width The *series* is the combination of weight and width. A font's *weight* is the thickness of the strokes and the *width* is how wide the characters are. The Computer Modern family includes **bold fonts**.

¹Footnotes are easy to place.

Size Most L^AT_EX documents are typeset with 10 point text unless otherwise instructed. Larger sizes are used for titles, section titles, and so on. Abstracts and footnotes are normally set in 8-point type.

Font family The collections of all sizes of a font is called a *font family*.

2.5.2 Document font families

In a document class, the style designer designates three document font families:

1. *Roman* (upright and serified) document font family
2. *Sans serif* document font family
3. *Typewriter style* document font family

and picks one of these (for documents, as a rule, the roman document font family) as the *document font family* or *normal family*. In all the examples in this book, the document font family is the roman document font family except for presentations which use sans serif. In standard L^AT_EX, the three document font families are Computer Modern roman, Computer Modern sans serif, and Computer Modern typewriter.

In this book, the roman document font family is Times, the sans serif document font family is Helvetica, and the typewriter style document font family is Computer Modern typewriter. (Examples are typeset in Computer Modern.)

The document font family (normal family) is the default font. You can always switch back to it with

```
\textnormal{...} or {\normalfont ...}
```

Section [A.3.1](#) lists these two commands and three additional pairs of commands to help you switch among the three basic document font families. It also shows the command pairs for the basic font shapes.

Command pairs

The font-changing commands of Section [A.3.1](#) come in two forms:

- A command with an argument, such as `\textrm{...}`, changes its argument. These are short commands, i.e., they cannot contain a blank line or a `\par` command.
- A command declaration, such as `\rmfamily`, carries out the font change following the command and within its scope; see Section [2.2.2](#).



Practical Tip 24. You should always use commands with arguments for small changes within a paragraph, because you are less likely to forget to change back to the normal font and do not have to worry about italic corrections; see Section [2.5.4](#).

For font changes involving more than one paragraph, use command declarations.

2.5.3 Shape commands

There are five pairs of commands to change the font shape:

- `\textup{...}` or `{\upshape ...}` switch to the upright shape
- `\textit{...}` or `{\itshape ...}` switch to the *italic shape*
- `\textsl{...}` or `{\slshape ...}` switch to the *slanted shape*
- `\textsc{...}` or `{\scshape ...}` switch to SMALL CAPITALS
- `\emph{...}` or `{\em ...}` switch to *emphasis*

The document class specifies how emphasis is typeset. As a rule, it is italic or slanted unless the surrounding text is italic or slanted, in which case it is upright. For instance,

`\emph{Rubin space}`

in the statement of a theorem is typeset as

[*the space satisfies all three conditions, a so-called Rubin space that ...*

The emphasis changed the style of Rubin space from italic to upright.



Practical Tip 25. Be careful not to interchange the command pairs. For instance, if by mistake you type `{\textit serif}`, the result is serif. Only the *s* is italicized since `\textit` takes *s* as its argument.

Practical Rule ■ Abbreviations and acronyms

For abbreviations and acronyms use small caps, except for two-letter geographical acronyms.

So Submitted to TUG should be typed as

Submitted to `\textsc{tug}`

Note that only the lowercase characters in the argument of the `\textsc` command are printed as small caps. `\textsc{TUG}` prints as TUG, not as TUG.

2.5.4 Italic corrections

The phrase

[when using a *serif* font

can be typed as follows:

when using a `{\itshape serif\}` font

The `\/` command before the closing brace is called an *italic correction*. Notice that `{\itshape M}M` typesets as *MM*, where the *M* is leaning into the M. Type `{\itshape M\/}M` to get the correct spacing *MM*. Compare the typeset phrase from the previous example with and without an italic correction:

┌ when using a *serif* font
└ when using a *serif* font

The shape commands with arguments do not require italic correction. The corrections are provided automatically where needed. Thus you can type the phrase when using a *serif* font the easy way:

when using a `\textit{serif}` font



Practical Tip 26. Whenever possible, let L^AT_EX take care of the italic correction.

2.5.5 Series

These attributes play a very limited role with the Computer Modern fonts. There is only one important pair of commands,

`\textbf{...}` `{\bfseries ...}`

to change the font to bold.

2.5.6 Size changes

L^AT_EX documents, as a rule, are typeset in 10 point type. The 11 point and 12 point type are often used for greater readability and some journals require 12 point—if this is the case, use the 12pt document class option; see Section 6.6.5. The sizes of titles, subscripts, and superscripts are automatically set by the document class, in accordance with the font size option.

If you must change the font size for some text—it is seldom necessary to do so in a document—the following command declarations are provided:

```
\Tiny \tiny \SMALL \Small \small
      \normalsize
\large \Large \LARGE \huge \Huge
```

See Section A.3.2 for a visual representation of these commands.

The command `\SMALL` is also called `\scriptsize` and the command `\Small` is also called `\footnotesize`. The font size commands are listed in order of increasing—to be more precise, nondecreasing—size.

2.6 Lines, paragraphs, and pages

When typesetting a document, L^AT_EX breaks the text into lines, paragraphs, and pages. Sometimes you may not like how L^AT_EX has chosen to lay out your text. There are ways to influence how L^AT_EX does its work and these are discussed in this section.

2.6.1 Lines

L^AT_EX typesets a document one paragraph at a time. It tries to split the paragraph into lines of equal width with balanced spacing. If it fails to do so successfully and a line is too wide, you get an `overfull \hbox` message, as discussed in Section 1.4.

Breaking lines

There are two forms of the line breaking command:

- The `\\` and `\newline` commands break the line at the point of insertion but do not stretch it.
- The `\linebreak` command breaks the line at the point of insertion and stretches the line to make it of the normal width.

The text following any of these commands starts at the beginning of the next line, without indentation. The `\\` command is often used, but `\linebreak` is rarely seen. The `\\` command has an important variant: `\\[length]`, where `length` is the interline space you wish to specify after the line break. For instance, `length` may be `12pt`, `.5in`, or `1.2cm`. Note how the units are abbreviated.

2.6.2 Paragraphs

Paragraphs are separated by blank lines or by the `\par` command. Error messages always show paragraph breaks as `\par`.

Indentation can be prevented with the `\noindent` command and can be forced with the `\indent` command.

2.6.3 Pages

There are two page breaking commands:

- `\newpage`, which breaks the page at the end of the line next completed but does not stretch it
- `\pagebreak`, which breaks the page at the point of insertion and stretches it to normal length

Text following either command starts at the beginning of the next page, indented.

2.7 Spaces

The judicious use of horizontal and vertical space is an important part of the formatting of a document. Fortunately, most of the spacing decisions are made by the document class, but L^AT_EX has a large number of commands that allow the user to insert horizontal and vertical spacing.

Remember that \LaTeX ignores excess spaces, tabs, and end-of-line characters in the source file. If you need to add horizontal or vertical space in the typeset file, then you must choose from the commands in this section.

 **Practical Tip 27.** Use them sparingly!

2.7.1 *Horizontal spaces*

When typing text, there are four commands that are often used to create horizontal space; three are shown between the bars in the display below:

<code>\,</code>	<code>\,</code>
<code>\quad</code>	<code>\quad</code>
<code>\qquad</code>	<code>\qquad</code>

The fourth is the `\,` command, producing a thin space. You have seen its first example in this book at the end of Section 1.1.4.

The `\hspace` command takes a length as a parameter. For example, `\textbar\hspace{12pt}\textbar` prints as | |. This command is ignored at the beginning of a line; use `\hspace*` instead.

The length can be negative. This is often used when placing illustrations. Negative thin space is provided by the `\!` command.

The `\hfill`, `\dotfill`, and `\hrulefill` commands fill all available space in the line with spaces, dots, or a horizontal line, respectively. If there are two of these commands on the same line, the space is divided equally between them. These commands can be used to center text, to fill lines with dots in a table of contents, and so on.

To obtain

2. Boxes.....	34
ABC	and DEF
ABC	and DEF

type

```
2. Boxes\dotfill 34\\
ABC\hfill and\hfill DEF\\
ABC\hrulefill and\hrulefill DEF
```

In a centered environment—such as the `center` environment—you can use `\hfill` to set a line flush right:

	This is the title	
		First Draft
	Author	

typed as

```
\begin{center}
  This is the title\\
  \hfill First Draft\\
  Author
\end{center}
```

2.7.2 Vertical spaces

You can add some interline space with the command `\[length]`, as discussed in Section 2.6.1. You can also do it with the `\vspace` command, which works just like the `\hspace` command; see Section 2.7.1. Here are some examples:

```
\vspace{12pt} \vspace{.5in} \vspace{1.5cm}
```

This command is ignored at the beginning of a page; use `\vspace*` instead.

Standard amounts of vertical space are provided by the three commands

```
\smallskip \medskip \bigskip
```

As a rule, they represent a vertical space of 3 points, 6 points, and 12 points, respectively.

The vertical analogue of `\hfill` is `\vfill`. This command fills the page with vertical space so that the text before the command and the text after the command stretch to the upper and lower margin.

2.8 Boxes

Sometimes it can be useful to typeset text in an imaginary box, and treat that box as a single large character. A single-line box can be created with the `\text` command.

2.8.1 Line boxes

The `\text` command provides a *line box* that typesets its argument without line breaks. As a result, you may find the argument extending into the margin. The resulting box is handled by \LaTeX as if it were a single large character. For instance,

```
\text{database}
```

causes \LaTeX to treat the eight characters of the word `database` as if they were one. This technique has two major uses. It prevents \LaTeX from hyphenating the argument; see Section 2.3.3. It allows you to use the phrase in the argument in a formula; see Section 4.3.3.

The argument of `\text` is typeset in a size appropriate for its use, for example, as a subscript or superscript. See Section 4.3.3 for an example.

2.8.2 *Marginal comments*

Do not
use this
much.

The `\marginpar` command allows you to add marginal comments. So

```
\marginpar{Do not use this much.}
```

produces the comment displayed in the margin. Marginal comments appear on the left on even-numbered pages and on the right on odd-numbered pages.

2.8.3 *Paragraph alignments*

Horizontal alignment of a paragraph is controlled by the `flushleft`, `flushright`, and `center` environments. Within the `flushright` and `center` environments, it is customary to force new lines with the `\\` command, while in the `flushleft` environment, you normally allow \LaTeX to wrap the lines.

There are command declarations that correspond to these environments:

- `\centering` centers text
- `\raggedright` left aligns text
- `\raggedleft` right aligns text

The effect of one of these commands is almost the same as that of the corresponding environment except that the environment places additional vertical space before and after the displayed paragraphs. For such a command declaration to affect the way a paragraph is formatted, the scope must include the whole paragraph, including the blank line at the end of the paragraph, preferably indicated with a `\par` command.

Practical LaTeX

Grätzer, G.

2014, XVI, 216 p. 55 illus., 20 illus. in color. With online
files/update., Softcover

ISBN: 978-3-319-06424-6