LaTeX for Word Processor Users
version 1.0.7

Guido Gonzato, Ph.D.
`guido.gonzato@univr.it`
Università di Verona (Italy)
Direzione Informatica

September 14, 2010

Abstract

Text processing with LaTeX offers several advantages over word processing. However, beginners may find it hard to figure out how to perform common tasks and obtain certain features. This manual attempts to ease the transition by drawing comparisons between word processing and LaTeX typesetting. The main word processor capabilities are listed, along with their equivalent LaTeX commands. Many examples are provided.

Contents

List of Tables

List of Figures

1   Introduction

First of all, let me state that this is *not* a LaTeX primer! If you're reading this document, I assume that you have at least a basic understanding of LaTeX and of its basic commands. In this guide, I'll explain how to replace a word processor effectively using LaTeX.

Word processors are the 'killer app' in modern office automation. They're perceived to be easier than LaTeX as they have a friendly WYSIWYG interface, and the average secretary will learn to use them in a relatively short time. The problem is, these beasts keep growing slow, bloated,[1] buggy, crash-prone, expensive, virus ridden, and incompatible with each other. Not to talk about their default output quality.

LaTeX is a good alternative; but if you try to write casual, unstructured documents, it will rightly stand in the way.

To sum up, sometimes you may want to use word processor-like features—but using LaTeX. It would be nice to know how to obtain some effects with LaTeX when you know how to

---

[1]once upon a time, I wrote my thesis on a 128k RAM, Z80-based home computer. The word processor WordStar and my thesis fit on a single CP/M-bootable 720K floppy, with lots of room to spare!

get them with your once-favourite `:-)` word processor.

That's why I wrote this quick reference. As I said, it assumes some basic LATEX knowledge; if it's not the case, I suggest that you go to http://www.ctan.org/starter.html and download 'The (Not So) Short Introduction to LATEX2e'. Another good primer is http://en.wikibooks.org/wiki/LaTeX/.

In the following sections, we shall navigate through the menus and menu items of an imaginary word processor, finding out the corresponding LATEX way of doing the same work. If you're a LATEX purist and this approach makes you sick... don't hurt yourself, please stop here.

1.1   Preliminaries

Many word processor features are implemented by the editor; others by standard LATEX commands; others still are obtained using *packages*. These are sets of macros that extend LATEX providing new commands and environments. There are lots of packages around: the only problem is knowing where they are, what they do, and how to install them. More about packages in Section 1.1.2.

Packages and other TEX-related material are available at many sites that constitute the CTAN: the Comprehensive TeX Archive Network. I already mentioned http://www.ctan.org; this site has a wide list of mirrors. From now on, CTAN: means 'your favourite CTAN mirror here, starting from the TEX directory'. For instance, you can get LATEX for your platform from CTAN://systems (e.g. http://www.tex.ac.uk/tex-archive/systems/).

To write your documents, you will need a good text editor. A better choice for beginners is a *LATEX shell*, i.e. an editor dedicated to writing LATEX source, with preview and many facilities.

I suggest that you install one of the programs listed below; all of them are Free/Open Source software.

- Texmaker (multiplatform):
  http://www.xm1math.net/texmaker/index.html

- TeXworks (multiplatform):
  http://tug.org/texworks/

- LyX, an almost-WYSIWYG LATEX editor (multiplatform):
  http://www.lyx.org/

- TeXShop (Mac OS X):
  http://www.uoregon.edu/~koch/texshop/

- TeXnicCenter (Windows):
  http://www.texniccenter.org/

A comprehensive list of LATEX shells for Windows can be found at http://home.arcor.de/itsfd/texwin. Information about LATEX on the Mac can be found at http://www.esm.psu.edu/mac-tex/.

1.1.1 Editor-Supported Features

LaTeX is just a formatter: things like cut and paste, search and replace etc. are delegated to the editor. Table 1 summarises the main commands of popular editors for geeks: GNU `emacs` and `vim` with their native key bindings, and `jed` configured for Borland IDE key bindings.

1.1.2 Adding Packages

The following information applies to `TeX Live`, which ships with most GNU/Linux distributions. It may also work on MacTeX, but I have no first-hand experience. Instructions for MiKTeX (probably the most popular implementation for Windows) will follow below.

A large collection of LaTeX packages is natively supported; for instance, Ubuntu provides lots of `texlive-*` packages. If you need to install an unsupported package, do the following.

1. create this directory structure:

   ```
   $ mkdir -p ~/texmf/tex/latex
   ```

   Under this directory, new packages will be installed.

2. get the package (typically as a zip-compressed directory) from your favourite CTAN mirror; let's call it `foo.zip`

3. unpack it in the right place:

   ```
   $ mkdir ~/texmf/tex/latex/foo
   $ mv foo.zip ~/texmf/tex/latex/foo
   $ cd ~/texmf/tex/latex/foo ; unzip foo.zip
   ```

4. If no `.sty` file exists, run the command `latex foo.ins` or `latex foo.dtx` to create it;

5. run the command `texhash  /texmf`.

To add a package to MiKTeX, create the directory `\latex\newpackage` under `C:\localtexmf\tex\` and put the relevant files in there. Proceed as above, then run MiKTeX Options and click on the 'Refresh now' button. Alternatively, issue the command `initexmf -u`. That's it!

Once a package is installed, you make it available in your documents adding a line under the `documentclass` declaration:

```
\usepackage{foo}
```

Action	Emacs	Vim	Jed
command mode	`Alt-X`	`ESC`	`Alt-X`
insert mode	n/a	`i a o O`	n/a
line editor mode	n/a	`:`	n/a
file operations			
open file	`Ctrl-X Ctrl-F`	`:e`	`Ctrl-KE`
insert file	`Ctrl-Xi`	`:r`	`Ctrl-KR`
save file	`Ctrl-X Ctrl-S`	`:w`	`Ctrl-KD`
save as	`Ctrl-X Ctrl-W name`	`:w name`	`Ctrl-KS`
close file	`Ctrl-XK`	`:q`	`Ctrl-KQ`
change buffer	`Ctrl-XB`	`bN`	`Ctrl-KN`
undo	`Ctrl-XU`	`u`	`Ctrl-U`
redo	`Ctrl-^`	`Ctrl-R`	`Ctrl-G Ctrl-U`
exit	`Ctrl-X Ctrl-C`	`:qa!`	`Ctrl-KX`
moving around			
word left	`Alt-B`	`b`	`Ctrl-A`
word right	`Alt-F`	`w`	`Ctrl-F`
start of line	`Ctrl-A`	`0`	`Ctrl-QS`
end of line	`Ctrl-E`	`$`	`Ctrl-QD`
page up	`Alt-V`	`Ctrl-U`	`Ctrl-R`
page down	`Ctrl-V`	`Ctrl-D`	`Ctrl-C`
start of buffer	`Alt-<`	`1G`	`Ctrl-QR`
end of buffer	`Alt->`	`G`	`Ctrl-QC`
line n.	`Alt-G n.`	`n.G`	`Ctrl-QI`
deleting			
character left	`Ctrl-H`	`X`	`BS`
character right	`Ctrl-D`	`x`	`Alt-G`
word left	`Alt-DEL`	`db`	`Alt-BS`
word right	`Alt-D`	`dw`	`Ctrl-T`
end of line	`Ctrl-K`	`d$`	`Ctrl-QY`
line	`Ctrl-A Ctrl-K`	`dd`	`Ctrl-Y`
search & replace			
search	`Ctrl-S text`	`/text`	`Ctrl-QS`
replace	`Alt-%`	`:s/old/new/g`	`Ctrl-QA`
blocks			
start selection	`Ctrl-SPACE`	`v`	`Ctrl-KB`
cut	`Ctrl-W`	`D`	`Ctrl-KY`
copy	`Alt-W`	`Y`	`Ctrl-KH`
paste	`Ctrl-Y`	`P`	`Ctrl-KC`

Table 1: Useful key bindings for Emacs, Vim, and Jed in IDE mode.

1.2   The Golden Rules

Before we start, please keep this in mind:

1. get used to *structuring* your documents: think in terms of parts, chapters, sections, and so on. This holds true even if you're not writing a scientific paper.

2. LaTeX explicitly discourages the user from messing about with formatting parameters. Don't worry too much about the *looks* of what you're writing, concentrate on the *contents*.

Convince yourself, apply the rules, and your printed material will magically look professional! Again, let me recommend that you study a real LaTeX primer.

These rules apply to most documents. However, if the stuff you are going to write is largely unstructured (circulars, notices, etc.), then you asked for that: go ahead and read the rest of this guide.

2   The File Menu

Obviously, some entries of this menu have nothing to do with LaTeX: File/Open, File/Save, File/Close depend on the editor.

2.1   File/New

This is the LaTeX equivalent of a blank page:

```
\documentclass{article}
\begin{document}
% This is a comment. Write your stuff here.
\end{document}
```

As documents written in LaTeX are inherently structured, this is a more realistic example:

```
\documentclass[a4paper,12pt]{article}
\begin{document}
\title{My Document}
\author{John Smith}
\date{London, \today}
\maketitle
\begin{abstract}
This is a very short article.
\end{abstract}
\tableofcontents
\listoftables
\listoffigures
\section{First Section}
```

```
\label{sec:start}
This is the text of the section. See \cite{Gonzato} for details.
\section{End}
\label{sec:end}
This is the end of the document. Please go to Section
\ref{sec:start} to read it again.
\begin{thebibliography}{99}
\bibitem{Gonzato} Gonzato G. \textit{\LaTeX{} for Word Processor
Users}. CTAN, 2001.
\end{thebibliography}
\end{document}
```

More document templates are listed in Appendix A.

2.2   File/Save As. . .

The following tools are handy if you want to convert LaTeX to other formats:

- TeX4ht is probably the best LaTeX to HTML converter:
  http://www.cse.ohio-state.edu/~gurari/TeX4ht/

- latex2html, another converter to HTML:
  http://saftsack.fs.uni-bayreuth.de/~latex2ht/,
  CTAN://support/latex2html

- latex2rtf, a converter to Rich Text Format:
  CTAN://support/latex2rtf

- detex (a command line converter) removes all LaTeX tags and outputs plain text:
  http://www.cs.purdue.edu/homes/trinkle/detex/,
  CTAN://support/detex/

See also Section 2.7 for details on PDF creation.

2.3   File/Save As Template

Saving a LaTeX 'template' would mean, as I see it, create a new LaTeX package. This is complex matter that goes beyond the scopes of this guide.

2.4   File/Import

These tools convert from other formats to LaTeX:

- rtf2latex: CTAN://support/rtf2latex

- html2latex: CTAN://support/html2latex

- wvware is a set of tools that convert from MS Word to several formats including LaTeX; http://wvware.sourceforge.net

- the free word processor Abiword, http://www.abisource.com, imports MS Word and can export LaTeX.

- `txt2tex`: CTAN://support/txt2tex does a fairly good job at converting plain text files into LaTeX.

Other `*2latex` converters are available at the same address.

Another interesting extension is OOoLatex, a set of macros for OpenOffice: http://ooolatex.sourceforge.net/.

2.5 File/Page Setup

The normal method for setting paper size, orientation, and margins, is to use parameters in `\documentclass`. Paper size can be `a4paper`, `a5paper`, `b5paper`, `letterpaper`, `legalpaper`, `executivepaper`; orientation is `portrait` by default, or optionally `landscape`. For example,

```
\documentclass[a5paper,landscape,12pt]{article}
```

Document-wide margins are set with these commands:

```
\setlength{\leftmargin}{2cm}
\setlength{\rightmargin}{2cm}
\setlength{\oddsidemargin}{2cm}
\setlength{\evensidemargin}{2cm}
\setlength{\topmargin}{-1cm}
\setlength{\textwidth}{18cm}
\setlength{\textheight}{25cm}
```

The geometry package allows for complete control of parameters such as paper size, margins width, and so on. geometry has far too many options to list them all, and you're invited to read its documentation. A fairly complete example of its usage is shown below. In this example, some parameters are not compatible with each other, and are there for purpose of argument only.

```
\usepackage{geometry}  % top of document
...
\geometry{paperwidth=25cm}
\geometry{paperheight=35cm}
% or: \geometry{papersize={25cm,35cm}}
\geometry{width=20cm}  % total width
\geometry{heigth=30cm} % total heigth
% or: \geometry{total={20cm,30cm}}
\geometry{textwidth=18cm}  % width - marginpar
\geometry{textheight=25cm} % heigth - header - footer
% or: \geometry{body={18cm,25cm}}
\geometry{left=3cm}    % left margin
\geometry{right=1.5cm} % right margin
% or: \geometry{hmargin={3cm,2cm}}
```

```
\geometry{top=2cm}     % top margin
\geometry{bottom=3cm}  % bottom margin
% or: \geometry{vmargin={2cm,3cm}}
\geometry{marginparwidth=2cm}
\geometry{head=1cm}    % header space
```

Options can also be set this way:

```
\usepackage[left=3cm, right=2cm]{geometry}
```

2.5.1   Page Setup/Headers and Footers

The fancyhdr package provides the new command \pagestyle{fancy}. This will create a header reporting the current section (chapter in book.cls) and subsection, and a footer with the page number: pretty fancy indeed. Headers and footers can obviously be customised. They are made of three parts: a left-aligned part, a centre-aligned part, and a right-aligned part. To set those, use the commands like in this example:

```
\usepackage{fancyhdr}
...
\lhead{} % empty
\chead{Hello, world!}
\rhead{Page \thepage} % page number
\lfoot{}
\cfoot{\textbf{Hello!}}
\rfoot{}
```

2.6   File/Printer Setup

This is highly OS-dependent, and definitely outside of the scopes of LaTeX. Assuming that you're using a UNIX-like system, these tips may come in handy:

- lpr -P printername prints to the specified printer;

- lpr -# 10 prints 10 copies;

- lpr -r removes the file after printing it

See below for more tips.

2.7   File/Print Preview

When the LaTeX input file is ready, you have several options:

- convert it to .dvi (latex file.tex) and previewing it with xdvi, yap or other DVI previewer;

- convert the .dvi to POSTSCRIPT with dvips, then use some variant of Ghostview;

- producing a .pdf file with dvipdf or directly with pdflatex.

In my opinion, producing a `.pdf` file is the best option, as it makes for widest portability.

While `dvipdf` is just a script that converts the `.dvi` to `.ps` then to `.pdf`, using `pdflatex` is more interesting. In fact, some packages like hyperref and url make the resulting `.pdf` file browseable; see Section 4.15. However, using `pdflatex` you might experience problems with other packages. More details in Section 4.12.

2.8   File/Print

Simply use `lpr file.ps` from the command line, or via the File/Print menu entry of your previewer.

2.9   File/Versions

Although the version package provides basic facilities for versioning LATEX sources, it's better to use external tools like RCS (Revision Control System) or CVS (Concurrent Version Control System), possibly integrated with the editor. For a quick introduction to CVS and RCS, please visit http://www.faqs.org/docs/Linux-HOWTO/CVS-RCS-HOWTO.html.

3   The Edit Menu

This menu has more to do with the editor than LATEX features. To begin with, refer to Table 1 that shows the key bindings for the entries Edit/Cut, Edit/Copy, Edit/Paste, Edit/Find, and Edit/Replace of some common editors.

Selecting text is not only performed for subsequent cut and paste, but also for applying styles to the selected text. The equivalent action in LATEX is to enclose the text either between braces, or in an environment. For example, to apply the bold attribute to a portion of text you will use one of the following:

```
this is \textbf{bold text;}\\
this is also
{\bfseries bold text;}\\
\begin{bfseries}
this is bold text, too!
\end{bfseries}
```

> this is **bold text;**
> this is also **bold text;**
> **this is bold text, too!**

3.1   Edit/Autotext

We call *autotext* the feature that makes you type, say, 'PS' to have the word 'PostScript' automatically expanded. This is an editor feature, but there's a rough LATEX equivalent:

```
\def\PS    {\textsc{PostScript}}
```

which will insert the equivalent of `\textsc{PostScript}` whenever you type `\PS`. Beware, case is important.

4   The **Insert** Menu

4.1   **Insert/Breaks**

- a non-breakable space is set with ˜ (tilde)

- to force a line break, use `\linebreak` or `\newline`; see below for details

- to start a new paragraph use a blank line, or `\\` (possibly followed by an optional space, as in `\\[1cm]`), or `\par`

- finally, for a page break use `\newpage` or `\clearpage`

Unlike `\newline`, `\linebreak` stretches the line to the margins:

`I am stretched!\linebreak`
`But I am not.\newline`
`Ok, now you get it.`
	I am stretched!
	But I am not.
	Ok, now you get it.

Besides, `\clearpage` differs from `\newpage` in that it prints all pending *floats*, that is figures and tables. Floats will be explained in Section 4.12.

4.2   **Insert/Enumerated List**

Bulleted and enumerated lists correspond to the `itemize` and `enumerate` environments. To customise the bullet in a list environment, you can specify it as an argument to `\item`:

`\begin{itemize}`
`  \item[*] with an asterisk;`
`  \item[-] with a dash;`
`  \item[.] with a dot.`
`\end{itemize}`
	* with an asterisk;
	
	  - with a dash;
	
	  . with a dot.

Another way is to redefine the style of the counters that refer to the digits in the first-to fourth-level lists.[2] There are several styles: `\arabic` for 'normal' numbers, `\roman` for lower-case roman numerals (e.g. viii is 8), `\Roman` for upper-case roman numerals, `\alph` and `\Alph` for lower- and upper-case letters, and `\fnsymbol` that I will not explain for now:

`\begin{itemize}`
`\renewcommand{\labelitemi}{*}`
`\renewcommand{\labelitemii}{-}`
`  \item first level, item 1`
`  \item first level, item 2`
`  \begin{itemize}`
`    \item second level, item 1`
`    \item second level, item 2`
`  \end{itemize}`
`  \item first level, item 3`
`\end{itemize}`
	* first level, item 1
	
	* first level, item 2
	
	    - second level, item 1
	    - second level, item 2
	
	* first level, item 3

---

[2]These are `\labelitemi`, `\labelitemii`, `\labelitemiii` and `\labelitemiv`.

To use, say, roman numbers and uppercase letters in enumerated lists, do this:

```
\begin{enumerate}
\renewcommand{\labelenumi}
  {\Alph{enumi}}
\renewcommand{\labelenumii}
  {\roman{enumii}}
  \item first level, item 1
  \item first level, item 2
  \begin{enumerate}
    \item second level, item 1
    \item second level, item 2
  \end{enumerate}
  \item first level, item 3
\end{enumerate}
```

> A first level, item 1
>
> B first level, item 2
>
>   i second level, item 1
>
>   ii second level, item 2
>
> C first level, item 3

Alternatively, use the `enumerate` package. It redefines the `enumerate` environment with the ability to specify an optional argument. Any occurrence of one of the characters `A a I i 1` will produce the value of the counter, using (respectively) `\Alph`, `\alph`, `\Roman`, `\roman`, or `\arabic`. To include some text, enclose it in braces:

```
\begin{enumerate}[{Example} I.]
  \item First example.\label{item:first}
  \item Second example.
  \item Last example.
    Go to Item~\ref{item:first}.
\end{enumerate}
```

> Example I. First example.
>
> Example II. Second example.
>
> Example III. Last example. Go to Item I.

Lists within paragraphs are implemented by the `paralist` package, which provides the `inparaenum` environment:

```
I'll throw in a list of items:
\begin{inparaenum}
  \item apples,
  \item pears, and
  \item oranges.
\end{inparaenum}
The same list can be labelled
with letters:
\begin{inparaenum}
  [\itshape a) \upshape]
  \item apples, \label{first}
  \item pears, and
  \item oranges. The first item is \ref{first}.
\end{inparaenum}
```

> I'll throw in a list of items: 1. apples, 2. pears, and 3. oranges. The same list can be labelled with letters: *a)* apples, *b)* pears, and *c)* oranges. The first item is a.

As seen above, the characters `A a I i 1` can be used to modify the counter. `paralist` can do much more, and I suggest that you read its documentation.

4.3   Insert/Special Character

First of all, let's remind that some characters have a special meaning in LATEX. They must be entered either with a leading \, or using them in math mode, or even writing special commands: see Table 2.
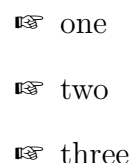
Character	LaTeX Sequence
$	`\$` or `\textdollar`
&	`\&`
%	`\%`
_	`\_` or `\textunderscore`
{	`\{` or `\textbraceleft`
}	`\}` or `\textbraceright`
<	`$<$` or `\textless`
>	`$>$` or `\textgreater`
\	`\textbackslash`
|	`\textbar`
•	`\textbullet`
‡	`\textdaggerdbl`
†	`\textdagger`
¶	`\textparagraph`
§	`\textsection`
©	`\textcopyright`
ˆ	`\textasciicircum`
˜	`\textasciitilde` or `\~{}`
∼	`$\sim$`
®	`\textregistered`
TM	`\texttrademark`
a	`\textordfeminine`
o	`\textordmasculine`

Table 2: How to obtain some special characters.

Another way to enter special characters is using their ASCII code and the `\char` command. For example, you could insert the characters `$&^~` entering `\char36 \char38 \char94 \char126`.

There are packages that provide thousands of unusual characters and symbols. For instance, pifont provides the commands `\ding`, `\dingfill`, `\dingline`, and `\dinglist`. The first command produces the Dingbat character of the specified code. The other commands are equivalent to the `\fill`, `\line`, and `\list` commands and environment, but use the Dingbat code given as parameter:
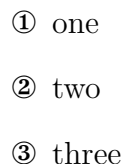
```
\begin{dinglist}{43}
  \item one
  \item two
  \item three
\end{dinglist}
```

☞ one

☞ two

☞ three

Another nice variant is this:

```
\begin{dingautolist}{172}
  \item one
  \item two
  \item three
\end{dingautolist}
```

① one

② two

③ three

There are far too many symbols to mention them all in this guide. Rather, I'll point you to 'The Comprehensive LaTeX Symbol List' at CTAN://info/symbols/comprehensive.

4.3.1 The € Sign

The official Euro sign is provided by the eurosym package, which can be used in two ways:

```
\usepackage[gen]{eurosym}
\usepackage[official]{eurosym}
```
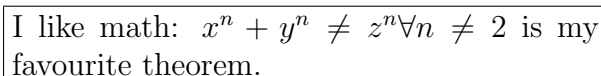
Both provide the `\euro` command, which produces €. The actual shape of the symbol depends on the declaration: the `[gen]` option provides a symbol that works with all font styles; the second declaration produces €. Please note the difference. The latter shape can always be obtained with `\officialeuro`.

Another package that provides the Euro sign is marvosym, which also provides many more fine characters. You get € with `\EUR`.

4.4 Insert/Formula

LaTeX is particularly strong at typesetting math. To insert math symbols in the main text, you must enclose them between `$`:

```
I like math: $x^n + y^n \neq
z^n \forall n \neq 2$
is my favourite theorem.
```

I like math: $x^n + y^n \neq z^n \forall n \neq 2$ is my favourite theorem.

The environments `displaymath` and `equation` typeset formulae aside from the text. The latter adds an equation number for later reference:

```
Fermat's Last Theorem is
defined as:
\begin{equation}
x^n + y^n \neq z^n
\forall n \neq 2
\label{eq:fermat}
\end{equation}
Can you prove
Eq.~\ref{eq:fermat}?
```

> Fermat's Last Theorem is defined as:
> $$x^n + y^n \neq z^n \forall n \neq 2 \qquad (1)$$
> Can you prove Eq. 1?

4.5 Insert/Footnote

The command `\footnote[n]{footnote text}` is all you need; the optional parameter `[n]` modifies the footnote number. The `\footnote` command should be placed after commas, full stops, and other punctuation signs that follow the word.

To use a symbol or arbitrary text instead of a number, redefine the counter associated with `\footnote`:

```
\renewcommand{\thefootnote}{read me!}
This footnote\footnote
{I mean this one.}
says it all.
```

> This footnote$^{\text{read me!}}$ says it all.
> _____
> $^{\text{read me!}}$I mean this one.

Using this method, you can get footnote numbers in roman numerals, or replaced by nice symbols:

```
\renewcommand{\thefootnote}
{\Roman{footnote}}
This\footnote{The first.}
is the first footnote,
and this\footnote{The second.}
is the second.
\renewcommand{\thefootnote}
{\fnsymbol{footnote}}
The end.\footnote[8]{At last!}
```

> This$^{\text{II}}$ is the first footnote, and this$^{\text{II}}$ is the second. The end.†
> _____
> $^{\text{II}}$The first.
> $^{\text{II}}$The second.
> †At last!

Note the `\fnsymbol{footnote}` thing. It uses 9 symbols associated with the values $1 \ldots 9$ of the `footnote` counter: $* \quad \dagger \quad \ddagger \quad \S \quad \P \quad \| \quad ** \quad \dagger\dagger \quad \ddagger\ddagger$.

To make several references to the same footnote, don't write its number explicitly. Rather, do this:

```
This\footnote{the first.}
\newcounter{\myfootnote}
\setcounter{\myfootnote}
  {\value{footnote}}
and that\footnote{the second.}
are footnotes: please read note
\footnotemark
[\value{\myfootnote}] again.
```

> This^{1} and this^{2} are footnotes: please read note^{1} again.
> _____
> ^{1}the first.
> ^{2}the second.

Warning: minipages use their own counters, `mpfootnote` and `thempfootnote`.

4.5.1   Footnotes at End of Document

The endnotes package lets you move all footnotes at the end of the document. You'll have to add this line to the preamble:

```
\let\footnote=\endnote
```

and these lines as the last thing in your document:

```
\newpage
\begingroup
\parindent 0pt
\parskip 2ex
\def\enotesize{\normalsize}
\theendnotes
\endgroup
```

Other commands are available, please read the endnotes.sty source.

4.6   Insert/Indices

Generating and inserting a table of contents, list of tables, and list of figures is a trivial task in LaTeX. All you have to do is insert these lines before the first \section or \chapter of your document:

```
\tableofcontents
\listoffigures
\listoftables
```

4.7   Insert/Vertical and Horizontal Space

This entry doesn't actually exist in any word processor I am aware of. This is in fact a limitation that LaTeX fills in a very elegant way.

Space filling is used to center text horizontally, vertically, or both; this is a difficult task to perform with any word processor, and requires a lot of trial end error. Use a combination of \null or ~ to set fixed marks, followed by \vfill and \hfill like in this example:

```
one \hfill two\\
\vfill
~ \hfill three \hfill ~\\
\vfill
four \hfill five
\null
```

one	two
	three
four	five

Normally, LaTeX won't let you insert blank spaces at your will. However, if you do want to make your document look messy, use ~ to make a non-breakable space.

Also, use the command \hspace like in this example:

```
This is a \hspace{2cm}
2-cm-wide hole.
```

This is a	2-cm-wide hole.

4.8 Insert/Tabs

The `tabbing` environment provides a rough equivalent to the action of the TAB key. See this example:

```
\begin{tabbing}
% let's set the tab positions
~ \hspace{1cm} \= ~ \hspace{2cm} \=
~ \hspace{3cm} \= \kill % discard text
Zero \> One \> Two \> Three \\
Zero \> One \> \> Three \+ \\ % go right
Zero \> Two \> Three \- \\ % go left
Zero \> One \> Two \\
\pushtabs % save tab positions
new tab 1{\ldots} \= new tab 2 \\
new \> tab \\
\poptabs  % restore tab positions
Zero \> One \> Two \> Three
\end{tabbing}
```

Zero	One	Two	Three
Zero	One		Three
	Zero	Two	Three
Zero	One	Two	
new tab 1...	new tab 2		
new		tab	
Zero	One	Two	Three

See also the `tabular` and `table` environments.

4.9 Insert/Cross Reference

The commands `\label`, `\ref`, and `\pageref` are all you need to insert labels in the text and do cross referencing. The standard format of labels is the `prefix:suffix` form, where `prefix` is one of the following: `cha` for chapters, `eq` for equations, `fig` for figures, `sec` for (sub)sections, and `tab` for tables.

References to a page (section, table, figure, etc.) number can be obtained using `\label` and `\ref` as in this example:

```
\paragraph{Example.}
\label{par:example}
This paragraph appears
in Section~\ref{par:example}
on page \pageref{par:example}.
```

> **Example.** This paragraph appears in Section 4.9 on page 16.

Of course, you may use your own prefixes. For example, take this enumerated list:

```
\begin{enumerate}
  \item{first step: skip to
  \ref{item:end} \label{item:start}}
  \item{another step (unreferenced)}
  \item{end: go back to
  \ref{item:start} \label{item:end}}
\end{enumerate}
```

> 1. first step: skip to 3
>
> 2. another step (unreferenced)
>
> 3. end: go back to 1

4.10 Insert/Margin Notes

Very simple: use `\marginpar{text}`.

4.11   Insert/Frame

Let's imagine you want to typeset a poster or a sign: you'll need a way to put text and figures on fixed position on the page. Use the textpos package, as shown in the example listed in Figure 8 (see Appendix A).

> A simpler approach is using *minipages*: literally, miniature pages. This text is included in a minipage environment. Actually, to be more precise, a boxedminipage, courtesy of the boxedminipage package. I used this declaration:
>
> ```
> \begin{boxedminipage}[c]{0.6\linewidth}
> ... text ...
> \end{boxedminipage}
> ```

4.12   Insert/Figure

(The reference guide for graphic inclusion in LaTeX is 'Using Imported Graphics in LaTeX2e', a.k.a. epslatex.ps.)

A 'figure' can be not only a picture, but also a portion of text, a table, etc. that you put in a figure environment. This is an example:

```
\begin{figure}[htbp]
% [htbp] specifies the
% preferred placement: here, top,
% bottom, or separate page.
  \begin{center}
  \texttt{=8-)}
  \end{center}
  \caption{A smiley representing
  the author of this guide.}
  \label{fig:mysmiley}
\end{figure}
```

=8-)

Figure 1: A smiley representing the author of this guide.

Please note that figures are not guaranteed to appear exactly where you write the code! In fact, the main difference with word processors is that figures don't have a fixed placement; they 'float' to the optimal position that LaTeX finds for them. So, the text shouldn't refer to a figure like 'the figure below' or 'the figure above'; rather, use 'see Figure~\ref{fig:label}'.

Owing to this property, figures and tables are called *floats*. If you do need to position a float exactly, use the here package that provides an optional placement argument H.

Given a picture in Encapsulated POSTSCRIPT (.eps) format, you insert it in a LaTeX source file using the graphicx package and commands like those shown in Figure 2.

When you typeset your document with \latex then dvips, graphic inclusion only works with EPS files; pdflatex accepts JPG, PNG, and of course PDF files.

There are several packages that convert common graphic formats like .jpg, .gif, .png etc. to .eps; for example, ImageMagik (http://www.imagemagik.org) and The GIMP (http://www.gimp.org). However, these applications produce huge POSTSCRIPT files.

```
\begin{figure}
\begin{center}
\fbox{\includegraphics
[width=0.5\textwidth, angle=-90]
{gnuplot.ps}}
\caption{A Gnuplot graph.}
\label{fig:gnuplot}
\end{center}
\end{figure}
```
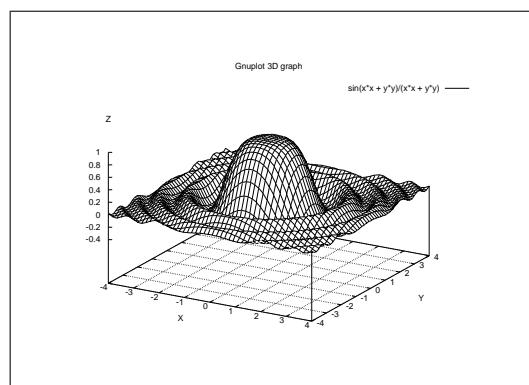

Figure 2: A Gnuplot graph.

Best results are obtained using applications that wrap the bitmap, turning it into a compact POSTSCRIPT file. You'll want to use `jpeg2ps` (http://www.pdflib.com/jpeg2ps/index.html) or `bmeps` (CTAN://support/bmeps). The former is often the best choice for wrapping `.jpg` files, but the latter handles more graphics formats.

☞   If you use PDFLATEX, you must convert your pictures to `.pdf` with the command
       `epstopdf` and modify the source accordingly!

If you wish to make both `.pdf` and `.ps` from the same source file, include these commands:

```
\usepackage{ifpdf}
...
% include the right options
\ifpdf
  \usepackage[pdftex]{graphicx}
  \pdfcompresslevel=9
\else
  \usepackage{graphicx}
\fi
...
% include the right graphic file
\ifpdf
  \includegraphics{file.pdf}
\else
  \includegraphics{file.eps}
\fi
```

☞   If you have more than 18 figures without text between them, you'll get the 'Too
       many unprocessed floats' LATEX error. The quickest way to solve this problem is to
       put `\clearpage` after three or four figures.

4.12.1   Wrapping Floats

For a magazine-like layout, use the wrapfig package:

If you meet this guy, give him some money.

```
\begin{wrapfigure}[4]{l}[5pt]{2cm}
{\Huge
 \texttt{=8-)}
}
\end{wrapfigure}

The reason may not be apparent to you,
but I can assure that your money
will end up in good hands.
I say again, if you meet this guy,
give him some money: he knows how to
use it properly. OK?
```

> If you meet this guy, give him some money.
> =8-) The reason may not be apparent to you, but I can assure that your money will end up in good hands. I say again, if you meet this guy, give him some money: he knows how to use it properly. OK?

The parameters are the number of lines to be narrowed, the figure placement, the overhang, and the figure width.

4.13   Insert/Shapes

LaTeX provides a `picture` environment whithin which you use commands like `\circle`, `\oval` and so on. In my opinion, drawing pictures without a graphical environment is just too hard, and `picture` has several limitations too. It's much better to use the drawing program Xfig (http://www.xfig.org), only available for Unix. It has a couple of good tricks in store.
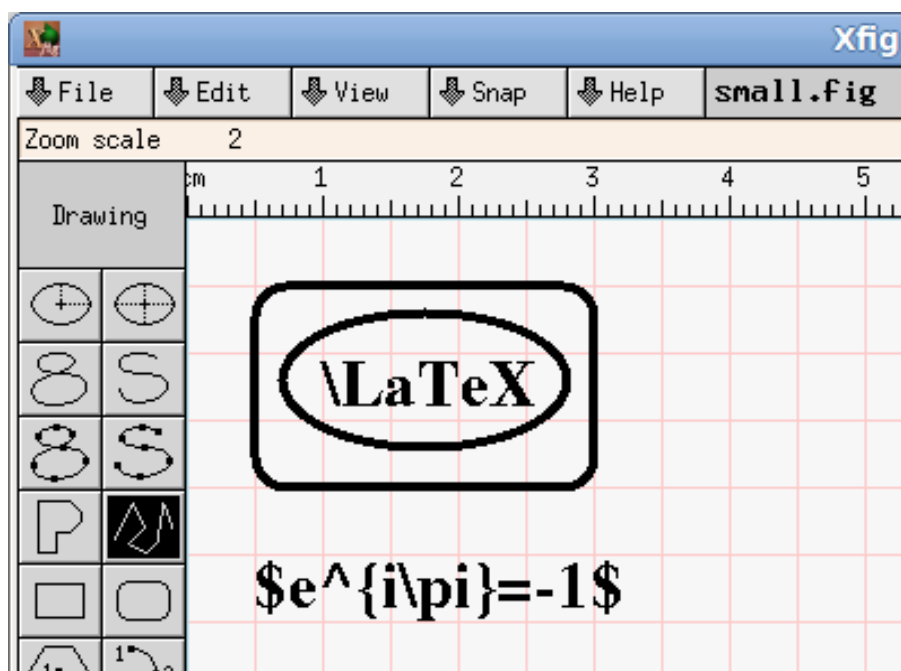

Figure 3: A quick drawing made with Xfig.

Xfig looks ugly, but is very powerful. One of its greatest advantages is that it exports drawings in several formats, some of which are recognised by LaTeX. Another bonus is that text objects will be rendered by LaTeX if their 'special flag' field is set, letting you enter ordinary TeX formulae and symbols in drawings.

So, let's suppose you made a drawing called `small.fig` (Figure 3). If you're going to use `pdflatex`, select File/Export… and select 'Combined PDF/LaTeX (both parts)' from the `Language` menu. Xfig will create two files, `small.pdf` and `small.pdf_t`. To include the drawing in a document, you'll do:

```
\usepackage[pdftex]{graphicx}
...
Here's an Xfig drawing:

\input{small.pdf_t}
```

Here's an Xfig drawing:

$$e^{i\pi} = -1$$
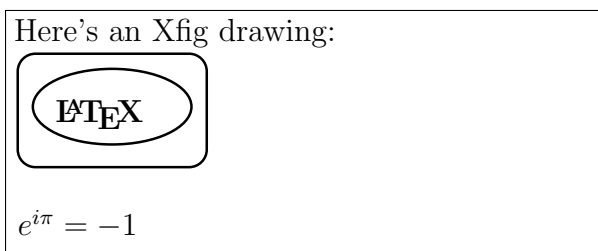
If you're going to use plain `latex`, select File/Export… and choose 'LaTeX picture + eepic macros' from the `Language` menu. You'll obtain another file, `small.eepic`. To include the drawing in a document, you'll have to use the epic and eepic packages:

```
\usepackage{epic}
\usepackage{eepic}
...
This is a picture
drawn with Xfig:\\
\input{small.eepic}
```

There's a caveat though: it looks like epic and eepic clash with hyperref, as the `\path` command is redefined by the latter. Another good reason to use `pdflatex`.

If you wish to do real magic, then check out the pgf package, http://sourceforge.net/projects/pgf/. Or pstricks: http://tug.org/PSTricks/main.cgi/. These packages let you make fine POSTSCRIPT drawings in LaTeX. Another wonderful program is ePiX (http://mathcs.holycross.edu/~ahwang/current/ePiX.html), specialised in producing scientific plots and figures for inclusion in LaTeX documents. Many more are available; search the web for "LaTeX vector graphics".

4.14   Insert/Line

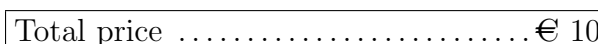Draw lines of any length and thickness with `\rule`:

```
This is a page-wide
rule:\\
\rule{\linewidth}{1pt}
but this one is shorter
and thicker:\\
\rule{2cm}{2mm}
```

This is a page-wide rule:

but this one is shorter and thicker:
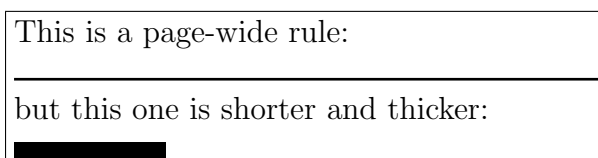
Another interesting 'line' is that made of dots (`\dotfill`), often used to relate things. This is how it's done:

```
Total price \dotfill \euro~10
```

Total price …………………………€ 10

4.15   Insert/Hyperlink

The url package lets you write URLs and have them hyphenated correctly. When used together with the package hyperref and dvipdf or pdflatex, url lets you make browseable .pdf documents! For instance, this document uses this declaration:

```
\usepackage[colorlinks,urlcolor=blue,filecolor=magenta]{hyperref}
\usepackage{url}
```

Let's see an example:

```
The \hypertarget{ctan}{CTAN} main site
is \url{http://www.ctan.org}, a.k.a
\href{http://www.ctan.org}{CTAN://}.

Listen to \href{run:midifile.mid}
{this MIDI file}.

Click \hyperlink{ctan}{here} to go
back to the top.
```

> The CTAN main site is http://www.ctan.org, a.k.a CTAN://.
> Listen to this MIDI file.
> Click here to go back to the top.

The \hypertarget and \hyperlink commands provide internal links, just like HTML; \href creates links to URLs or external files. Note the run: parameter: you can run external programs like multimedia players, office applications, whatever. As far as I know, this feature only works in Adobe Reader, Okular, and Evince.

On Linux and possibly other UNIX variants, you'll have to instruct your favourite PDF reader what to run when an external file is referenced. Insert lines like the following in your .mailcap or /etc/mailcap:

```
audio/midi;/usr/bin/timidity %s
audio/*; xmms %s
video/*; xine -pfhq %s
```

Please read hyperref's documentation for further examples and possibilities.

4.16   Insert/Comment

This is done inserting % before each line, or by using the package comment that provides the environment of the same name.

5   The Format Menu

In general, the main format properties of a document are set with parameters in \document-class: default font size (10, 11, or 12pt), paper (a4paper, a5paper, b5paper, letterpaper, legalpaper, executivepaper), and orientation (portrait, landscape). For example,

```
\documentclass[a5paper,landscape,12pt]{article}
```

Alternative font sizes can be specified as explained in Section 5.2.2.

5.1 Format/Line Spacing

The package setspace provide the environments singlespace, onehalfspace, and double-space. In addition, the environment/command \spacing{*amount*} will set the spacing to the specified amount:

```
\begin{spacing}{2.5}
These two lines \\
are crazily spaced!
\end{spacing}
\begin{spacing}{1}
Much better, these lines\\
have a pretty space.
\end{spacing}
```

> These two lines
>
> are crazily spaced!
>
> Much better, these lines
> have a pretty space.

To apply line spacing to the whole document, use the \linespread{*factor*} command in the preamble. Default value of factor is 1; larger values give larger line spacing (1.6 is roughly double line spacing).

5.2 Format/Character

Standard character properties are listed in Table 3, font sizes in Table 4.

Text attribute	Environment form	Example
\textnormal	textnormal	main document font
\textrm	rmfamily	roman
\textit	itshape	*italics*
\emph	n/a	*emphasis*
\textmd	mdseries	medium weight (default)
\textbf	bfseries	**boldface**
\textup	upshape	upright (default)
\textsl	slshape	*slanted*
\textsf	sffamily	sans serif
\textsc	scshape	SMALL CAPS
\texttt	ttfamily	typewriter
\underline	underline	underline
\textsuperscript	n/a	this is $^{\text{superscript}}$
\mathrm	n/a	$\mathrm{x^n + y^n \neq z^n \forall n \neq 2}$
\mathbf	n/a	$\mathbf{x^n + y^n \neq z^n \forall n \neq 2}$
\mathsf	n/a	$\mathsf{x^n + y^n \neq z^n \forall n \neq 2}$
\mathtt	n/a	$\mathtt{x^n + y^n \neq z^n \forall n \neq 2}$
\mathit	n/a	$\mathit{x^n + y^n \neq z^n \forall n \neq 2}$
\mathnormal	n/a	$x^n + y^n \neq z^n \forall n \neq 2$
\mathcal	n/a	$\S^\backslash + \dagger^\backslash \neq \ddagger^\backslash \forall^\backslash \neq \in$

Table 3: Font attributes.

Please note the difference between italics and emphasised text. *For example, this portion of text is typeset in italics, and* these words *are emphasised in upright.* As you can see, \emph is a *logical* rather than typographic command.

Also, please note that underscript is normally used in math mode only. The trick to use it in normal text is:

```
this is
$_{\mbox{\footnotesize{subscript}}}$
```

this is _{subscript}

Font size	Example
`tiny`	sample text
`scriptsize`	sample text
`footnotesize`	sample text
`small`	sample text
`normalsize`	sample text
`large`	sample text
`Large`	sample text
`LARGE`	sample text
`huge`	sample text
`Huge`	sample text

Table 4: Font sizes

5.2.1 Underline styles

Normally, underline is not used. It's just a relic of the old teletype era, and it doesn't look really good. If you still want to use underline, the ulem package provides some fancy styles:

```
\uline{important}
\uuline{urgent}
\uwave{boat}
\sout{wrong}
\xout{removed}
```

important urgent boat ~~wrong~~ removed

Beware: ulem redefines the `\emph` command, which will be replaced by underline. To avoid this behaviour, use this declaration:

```
\usepackage[normalem]{ulem}
```

5.2.2 Format/Character Size

If the standard font sizes aren't enough for you, the package extsizes may be handy. It provides 'extended' versions of the standard document classes, with support for sizes 8–12, 14, 17, and 20 pt.

For example, let's suppose you want to typeset an article using a 17 pt font. You'll use this document preamble:

```
\documentclass[17pt]{extarticle}
```

Family	Name
cmr	Computer Modern Roman
cmss	Computer Modern Sans Serif
cmtt	Computer Modern Typewriter
pag	Avantgarde
pbk	Bookman
phv	Helvetica
pnc	New Century Schoolbook
ppl	Palatino
ptm	Times
pcr	Courier

Table 5: Common font families.

Another way to get big fonts is to use the package type1cm, which provides commands like the following:

```
\fontsize{72pt}{72pt}\selectfont
No Smoking
```

(The example above is way too large to fit on this page. . .)

Parameters are font size and baseline. Yet another approach is this:

`\resizebox{!}{1cm}{1-cm tall}`   1-cm tall

5.2.3   Format/Character Font

LATEX uses its own fonts, automatically generated when needed by the METAFONT subsystem. This ensures portability and yields very good results. However, many of us are accustomed to other fonts: Times, Helvetica, Sans Serif. . .

Fortunately, LATEX can use POSTSCRIPT fonts. Try using one of the following packages: avant, avangar, bookman, chancery, charter, courier, helvet, helvetic, ncntrsbk, newcent, palatcm, palatino, pifont, times, utopia, zapfchan. Insert `\usepackage{times}` and enjoy the results. The only caveat is that LATEX handles maths at its best only with Computer Modern fonts: using POSTSCRIPT fonts might render your formulas slightly less appealing.

The packages above set the font for the whole document. To use a POSTSCRIPT font for a region of text only, specify the font family as in the example below. Common font families are listed in Table 5.

☞   Beware, some font shapes may be unavailable on some systems!

```
This is Computer Modern Roman,
{\fontfamily{phv}\selectfont
this is Helvetica!}
```
This is Computer Modern Roman, this is Helvetica!

Yet another possibility is replacing a standard LaTeX font with a POSTSCRIPT one: for example, you may want to use Avantgarde whenever Computer Modern Sans Serif would appear. These commands can be renewed as in the example below:

- `\rmdefault` (roman)

- `\sfdefault` (sans serif)

- `\ttdefault` (typewriter)

- `\bfdefault` (boldface)

- `\mddefault` (medium)

- `\itdefault` (italics)

- `\sldefault` (slanted)

- `\scdefault` (small caps)

- `\updefault` (upright)

```
 % Avantgarde replaces sans serif
\renewcommand{\sfdefault}{pag}
```

5.2.4   Format/Character Colour

You can colour words using the package color and appropriate commands. Predefined colours are black, white, red, green, blue, cyan, magenta, and yellow; you can also define your own.

```
\textcolor{red}{This is red.}\\
\color{blue}
This text is blue!\\
So is this. Let's change.\\
\definecolor{mygreen}
{rgb}{0.1,1,0.1}
\color{mygreen}
This is my shade of green!\\
\color{black}
\colorbox{cyan}{A cyan box}\\
\fcolorbox{blue}{green}
{A green box in a blue frame}
```

This is red.
This text is blue!
So is this. Let's change.
This is my shade of green!
A cyan box
A green box in a blue frame

Moreover, the command `\pagecolor` lets you specify... guess what?

5.3   Format/Paragraph

Let's remind what a paragraph is according to LaTeX: a portion of text that either ends with \\, or is followed by a blank line.

Environments are LaTeX's way of specifying properties like text alignment or font selection for a given portion of text. It's like selecting text with the mouse, then choosing the

property you wish from a menu or clicking on a button. Another way is to enclose the text between brackets.

Environments have this general form:

```
\begin{environment}
...text goes here...
\end{environment}
```

For example, if you want to center a paragraph you'll use the `center` environment:

```
\begin{center}
this text is centered
\end{center}
```

this text is centered

Standard environments are listed in Table 6. In the following sections, I'll show you what to use and when.

5.3.1 **Format/Paragraph Horizontal Alignment**

By default, the text is justified. To get left–aligned, right–aligned or centered text, use the `flushleft`, `flushright` and `center` environments. The commands `\raggedright`, `\raggedleft`, and `\centering` are equivalent to their correspondent environments, but they do not start a new paragraph.

5.3.2 **Format/Paragraph Vertical Alignment**

The way paragraphs are separated is often puzzling to word processor users. *Empty lines and multiple spaces are treated like a single empty line or space.* The commands `\smallskip`, `\medskip`, and `\bigskip` provide some space between paragraphs.

If you need more space, use the command `\vskip{`*parameter*`}` as in this example:

```
These paragraphs will be
separated by 1.3 cm:\\
\vskip 1.3cm
there is a 1.3 cm gap above me.
```

These paragraphs will be separated by 1.3 cm:

there is a 1.3 cm gap above me.

Note that `\vskip` only works between paragraphs. What if you wanted to start a page after an additional margin of, say, 1.5 cm? You'll have to use `\null`, which sets a 'mark' in the text:

```
\null
\vskip 1.3 cm
This text comes after 1.3 cm...
```

This text comes after 1.3 cm...

Finally, the command `\vfill` is used to add empty lines between two paragraphs so that the second paragraph goes exactly to the bottom of the page. For example,

Environment	Purpose
array	Math arrays
center	Centered lines
description	Labelled lists
enumerate	Numbered lists
eqnarray	Sequence of aligned equations
equation	Displayed equation
figure	Floating figures
flushleft	Flushed left lines
flushright	Flushed right lines
itemize	Bulleted lists
letter	Letters
list	Generic list environment
minipage	Miniature page
picture	Picture with text, arrows, lines and circles
quotation	Indented environment with paragraph indentation
quote	Indented environment with no paragraph indentation
tabbing	Align text arbitrarily
table	Floating tables
tabular	Align text in columns
thebibliography	Bibliography or reference list
theorem	Theorems, lemmas, etc
titlepage	For hand crafted title pages
verbatim	Simulating typed input
verse	For poetry and other things

Table 6: Standard LaTeX environments.

```
  This appears at the top of
  the page{\ldots}
  \vfill
  {\ldots}and this at the bottom.
```

> This appears at the top of the page...
>
>
>
> ...and this at the bottom.

5.3.3 Format/Paragraph Margins

Normally, the margins are set for the whole document as seen in Section 2.5. Redefining them for a section of text will not work: if you want to set a paragraph's margins, you'll have to create a new environment like in the following example:

```
\newenvironment{margins}[2]
{
  \begin{list}{} {
    \setlength{\leftmargin}{#1}
    \setlength{\rightmargin}{#2}
  } \item }
{\end{list}}
```

Then you will use the new environment:

```
As you can see, this paragraph
has normal margins.
\begin{margins}{0.5cm}{1cm}
 But please note that this
 paragraph has custom margins.
\end{margins}
```

> As you can see, this paragraph has normal margins.
>
> But please note that this paragraph has custom margins.

5.3.4 Format/Paragraph Indentation

To set the amount of indentation of the first line of a paragraph, use the command `\parindent`. In the following example, we set a 1-cm indentation:

```
\setlength{\parindent}{1cm}
```

The commands `\indent` and `\noindent` allow/disallow indentation on the following paragraph. Finally, the distance between paragraphs is set by `\parskip`:

```
\setlength{\parskip}{3pt}
```

5.4 Format/Paragraph Border and Shade

To get framed (bordered) paragraphs or words, you have the choice of using the framed package or the `\parbox` command. The package calc is required in the latter case.

This is the simplest method, using framed:

```
\setlength{\FrameRule}{2pt}
\setlength{\FrameSep}{5pt}
\begin{framed}
  this is a framed paragraph!
\end{framed}
\definecolor{shadecolor}{rgb}
{0.9,0.8,1}
\begin{shaded}
  this is a shaded paragraph,
  do you like it?
\end{shaded}
```

this is a framed paragraph!

this is a shaded paragraph, do you like it?

Equivalently, use the boxedminipage package and the equally named environment. For those who want to know more: the commands

```
\framebox{
  \begin{minipage}[c]{\linewidth}
  text to be framed
  \end{minipage}
}
```

are functionally equivalent to the boxedminipage environment.

This example uses \parbox:

```
\noindent
\fbox{
  \parbox{\linewidth
    -2 \fboxsep -2 \fboxrule}
  {again, a framed paragraph!}
}
```

-2 again, a framed paragraph!

\linewidth sets the width of the minipage equal to that of the remaining text. Obviously, you can specify the width as you like.

Finally, to frame something adapting the frame to the width of the text:

```
this is a
\framebox[\width]{framed}
word
```

this is a framed word

Modifying the parameter, you can adjust the frame width:

```
this is another
\framebox[2\width][r]{framed}
word
```

this is another   framed word

Note that the second optional parameter specifies the alignment (to the right in this example).

5.5   **Format/Colour**

Now that you have a bordered paragraph, you'll want to set its colour too. Do this:

```
\colorbox{yellow}{
  \begin{minipage}
  {0.8\linewidth}
  I am a minipage, my colour
  is yellow!
  \end{minipage}
}
```

I am a minipage, my colour is yellow!

More about colours in Section 5.2.4.

5.6   **Format/Columns**

The commands `\twocolumn` and `\onecolumn` start a new page and set the number of columns; they can also be used as parameters in `\documentclass`. If this is not enough for you, the package **multicols** provides an environment of the same name. I could have set this section in two columns with these commands:

```
\columnseprule=1pt
\begin{multicols}{2}[\subsection{\entry{Format}{Columns}}]
The commands \cmd{twocolumn} ...
\end{multicols}
```

The space between columns is controlled by the parameter `\columnsep`, and the thickness of the rule between columns by `\columnseprule`. The text given as optional parameter in brackets is excluded from the environment.

6   **The Table** Menu

Quite a complex subject. . .   A *table* is a float (as explained in Section 4.12) that must fit on one page. It usually contains a `tabular` environment, even though other possibilities exist. By default, a table adjusts its width to match the width of its contents.

This is the general format of a table:

```
\begin{table}[htbp] % placement: here, top, bottom, separate page
% \begin{small}     % sets the table font
\begin{center}      % optional
% 4-column table; alignment is left, centered, right, fixed width
\begin{tabular}{|l|c|rp{4cm}|}
\hline                % horizontal line
\textbf{Left} & \textbf{Centre} & \textbf{Right} & \textbf{4 cm} \\
\hline
row 1, col 1 & row 1, col 2 & row 1, col 3 & row 1, col 4 \\
\cline{1-2}         % horizontal line spanning columns 1-2
row 2, col 1 & row 2, col 2 & row 2, col 3 & row 2, col 4 \\
\cline{1-2}
\multicolumn{2}{|c|}{spanning two columns} & row 3, col 3 &
row 3, col 4 \\
```

```
\cline{1-3}
row 4, col 1 & row 4, col 2 & row 4, col 3 & ~ \hfill right \\
% force a space with "\ "
row 5, col 1 & row 5, col 2 & row 5, col 3 & left \hfill ~ \\
row 5, col 1 & row 5, col 2 & row 5, col 3 &
~ \hfill centre \hfill ~ \\
\hline
\end{tabular}
\caption{A sample table.}
% labels are used for cross references;
% for example, "see Table~\ref{tab:sampletab}"
\label{tab:sampletab}
\end{center}
% \end{small}
\end{table}
```

Table 7 shows the result.

Left	Centre	Right	4 cm	
row 1, col 1	row 1, col 2	row 1, col 3	row 1, col 4	
row 2, col 1	row 2, col 2	row 2, col 3	row 2, col 4	
spanning two columns		row 3, col 3	row 3, col 4	
row 4, col 1	row 4, col 2	row 4, col 3		right
row 5, col 1	row 5, col 2	row 5, col 3	left	
row 5, col 1	row 5, col 2	row 5, col 3	centre	

Table 7: A sample table.

Sometimes, a table is too wide and won't fit on the page. In that case, the rotating package provides the new environment `sidewaystable`. Also, rotating makes it possible to rotate the contents of a cell by a specified angle. Finally, the tabularx package lets one specify tables of fixed width: the `X` column specifier indicates that a column can be spread as needed.

Here's an example:

```
\begin{sidewaystable}
  \begin{tabularx}{7.5cm}{|l|X|X|}
    \hline
    \textbf{normal} & \textbf{tilted} &
    \textbf{wider} \\
    \hline
    normal & \rotatebox{30}{I'm tilted!} &
    I'm wider \\
    \hline
  \end{tabularx}
\end{sidewaystable}
```

normal	tilted	wider
normal	*I'm tilted!*	wider

The standard `tabular` environment cannot span more than one page! There are some packages that overcome this limitation: you will want to try out longtable, supertabular, and xtab.

To enable colours in tables, you use the **colortbl** package:

```
Colour by row:\\\vskip 2mm
\begin{tabular}{|l|c|r|}
  \hline
  \rowcolor{cyan}
  one & two & three\\
  \rowcolor{green}
  one & two & three\\
  \rowcolor{yellow}
  one & two & three\\
  \hline
\end{tabular}
```



```
Colour by column:\\\vskip 2mm
\begin{tabular}
  {|>{\columncolor{cyan}}l|
  >{\color{red}
  \columncolor{green}}c|
  >{\columncolor{yellow}}r|}
  \hline
  one & two & three\\
  one & two & three\\
  one & two & three\\
  \hline
\end{tabular}
```


To conclude the subject, a neat little trick. If you think that writing LATEX tables is too complicated, you could be relieved by OpenOffice Calc and Calc2LaTeX. The former is the well-known free spreadsheet, while the latter is a plugin that that lets you turn a cell range into a LATEX table. Links: http://www.openoffice.org/, http://calc2latex.sourceforge.net/.

6.1 Line Spacing

A line adjusts itself to the height of the text it contains. To add some space *before* a line, the trick is to start it with a `\rule` of 0 length and specified height. To add space *after* a line, use \\ followed by optional space. Here is an example:

```
\begin{tabular}{lll}
one & two & three\\
0.3 centimeters & \textbf{after} &
  this line\\[0.3cm]
one & two & three\\
one & two & three\\
\rule{0pt}{1.2cm}1.2 centimeters &
  \textbf{before} & this line\\
\end{tabular}
```

6.2   Rule Width

```
\begin{tabular}{|lll|}
\hline
%\setlength{\arrayrulewidth}{5pt}
one & two & three\\
\hline
four & five & six\\
%\setlength{\arrayrulewidth}{1pt}
\hline
\end{tabular}
```

one	two	three
four	five	six

6.3   Aligning Numbers

A special case of a tabular environment is when we want to align numbers with respect to the decimal positions.

The simplest method is using the @ column specifier, which in practice is useful in tables containing only numbers. The column separator & is replaced by the decimal dot:

```
\begin{tabular}{r@{.}l}
 3&14159\\
 1&61803\\
 1&41421\\
 100&00000
\end{tabular}
```

3.14159
1.61803
1.41421
100.00000

Alternatively, use the **dcolumn** package, which adds the D column specifier. D has three arguments: the separator to use in the LaTeX source and in output (usually the same, '.'), and the number of digits to the right of the decimal place indicator. Optionally, the third argument can specify the number of digits to the left and to the right of the decimal place indicator, separated by a dot. Lastly, if the third argument is -1, the material of the column is centered around the separator.

All material in the table is typeset in math mode. To insert headings, you'll have to put the text in an \mbox.

```
\begin{tabular}{|D{.}{,}{4.2}|%
D{.}{.}{5}|D{.}{.}{-1}|}
\hline
\mbox{One} & \mbox{Two} &
\mbox{Three} \\
10.33 & 10.33 & 10.33 \\
1000 & 1000 & 1000 \\
5.1 & 5.1 & 5.1 \\
3.14 & 3.14159 & 3.14159 \\
\hline
\end{tabular}
```

One	Two	Three
10,33	10.33	10.33
1000	1000	1000
5,1	5.1	5.1
3,14	3.14159	3.14159

6.4   Using **slashbox**

This package add the \backslashbox command:

```
\begin{tabular}{|l|l|l|}
  \hline
  \backslashbox[2cm]{Lesson}{Date} &
  Monday & Tuesday \\
  \hline
  Stratigraphy & room A & room A \\
  Chemistry & room B & Lab $\alpha$ \\
  Physics & room C & Lab $\beta$ \\
  \hline
\end{tabular}
```

Date Lesson	Monday	Tuesday
Stratigraphy	room A	room A
Chemistry	room B	Lab α
Physics	room C	Lab β

6.5   Importing Data in LaTeX Tables

For many people, data files are the bread and butter of everyday's work. Most data files are simply ASCII text with columns of numbers, but some people use spreadsheets. Nearly all spreadsheet applications can export sheets in the ASCII-based .cvf file format; values are separated by the ';' character.

Converting a data file into a LaTeX table is quite a tedious process. The following script for UNIX will convert a datafile with an arbitrary number of columns to a table. It will also work on .cvf files.

```
#!/bin/sh

# dat2tex: converts tabular data to a tabular environment

if [ $# != 1 ]; then
  echo "Usage: $0 <datafile>"
  exit 1
fi

# is this a cvs file?
grep ";" $1 > /dev/null
if [ $? = 0 ]; then
  AWK="awk -F;"
else
  AWK=awk
fi

# ok awk, make my day
$AWK '{if (1 == FNR) { \
      printf "\\begin\{tabular\}\{"; \
      for (i = 1; i <= NF; i++) {printf "l"}; \
      printf "\}\n"
    }
    for (i = 1; i < NF; i++) \
      {printf $i" & "} printf $NF" \\\\ \n"} \
    END {printf "\\end\{tabular\}\n"}' $1

# end of dat2tex
```

7   The **Tools** Menu

7.1   **Tools/Mail Merges**

This useful and time-saving tool is implemented in LaTeX by the `textmerg` package. Let's consider a simple document, in which the name, surname, and title of people we're writing to may vary. The remaining text does not change.

We'll define three *fields*, which are the variable part of the text: `\Name`, `\Surname`, and `\Title`. Their values will be gathered from an external file, `data.dat`.

```
\documentclass{article}
\usepackage{textmerg}
\begin{document}
% let's declare the variable fields:
% \Void is for empty lines
\Fields{\Name\Surname\Title-\Void}
\Merge{data.dat}{%
Dear \Title{} \Surname,\\
may I call you \Name?\\
Yours,\\
\hspace{3cm}Guido\clearpage}
\end{document}
```

The fourth field, `\Void`, isn't really necessary and it's there for illustration. It's preceded by a minus sign, which indicates that it can be empty in the data file. Simply put, we want to separate the records using empty lines.

The file `data.dat` reads:

```
Guido
Gonzato
Dr.

Francesco
Mulargia
Prof.

Marie
Curie
Mme
```

That's it: the resulting output will contain the merged text, one page for each recipient.

7.2   **Tools/Labels**

If making mail merges was easy, making labels is even trivial. Let's suppose you want to make 20 equal labels on a 3×8 peel–off label sheet. The package to use, predictably, is called labels. In this example, we'll make 10 plain labels and 10 boxed labels:

```
\documentclass[a4paper,12pt]{article}
\usepackage{labels}
\LabelCols=3        % n. of columns of labels
\LabelRows=8        % n. of rows of labels
\LeftBorder=8mm     % borders of each label
\RightBorder=8mm
\TopBorder=5mm
\BottomBorder=5mm
\LabelGridtrue        % show the grid
\numberoflabels=10  % number of labels of each type to print
% the text of the label is specified by
% the \addresslabel[]{} macro:
\begin{document}
  \addresslabel[\large] % optional arguments
  {\textbf{Guido Gonzato}, Ph.D.\\
  \textsl{Linux system manager}}
  % now on to the boxed labels
  \boxedaddresslabel[\fboxsep=4mm\fboxrule=1mm]
  {\textbf{Guido Gonzato}, Ph.D.\\
  \textsl{Linux system manager}}
\end{document}
```

To make labels containing different addresses, you may use either an external file or insert the addresses in the main file:

```
\documentclass[a4paper,12pt]{article}
\usepackage{labels}
\LabelCols=3
\LabelRows=8
\LeftBorder=3mm
\RightBorder=3mm
\TopBorder=8mm
\BottomBorder=8mm
\LabelGridtrue
\begin{document}
% use either this environment:
\begin{labels}
  1$^{st}$ name
  1$^{st}$ address
  1$^{st}$ city, state, zipcode

  2$^{nd}$ name
  2$^{nd}$ address
  2$^{nd}$ city, state, zipcode

  3$^{rd}$ name
  3$^{rd}$ address
  3$^{rd}$ city, state, zipcode
\end{labels}
% or an external file containing exactly the same text:
% \labelfile{addresses.dat}
\end{document}
```

It is left to you to combine textmerg and labels!

7.3   Tools/Default Language

LATEX default language is English, but other languages are supported. By language support I mean the translation of terms like 'Chapter' or 'Index', correct hyphenation, and the possibility of inserting characters like 'ç' or 'é' directly via your keyboard. (The normal way being typing \c c and \'e.)

Your LATEX distribution contains a file called `language.dat` (usually `$TEXMF/tex/generic/config/language.dat` that contains a list of languages. Editing this file you choose the languages for which you want hyphenation patterns.

If you are not a native English speaker, you'll want to use the package babel as in the following example:

```
\usepackage[italian,english]{babel}
```

☞   babel alters the way some characters behave in a language-dependent way. If you experience odd problems, insert the offending characters using the \charXX syntax.

In addition, to type accented letters and in general non-standard ASCII characters[3] you may want to use the package isolatin1. However, this is not a recommended practice, because it reduces the legibility and portability of your file: you had better stick to the TEX way.

If you just can't stand typying all those three- or four-letter clusters, then you could configure your editor to type those for you. For example, I set up my editor of choice (jed) to have it insert \'e whenever I type 'é'. I included this in my .jedrc:

```
define latex_mode_hook ()
{
  set_abbrev_mode (1);
  if ( () = abbrev_table_p ("LaTeX") )
    use_abbrev_table ("LaTeX");
#ifdef WIN32
  % prevent clash with movement keys
  undefinekey ("àà", "LaTeX-Mode");
  definekey (" \\`a", "àà", "LaTeX-Mode");
#else
  local_setkey (" \\`a",     "à");
#endif
  local_setkey (" \\'e",     "é");
  local_setkey (" \\`e",     "è");
  local_setkey (" \\'\\i{}", "ì");
  local_setkey (" \\`o",     "ò");
  local_setkey (" \\`u",     "ù");
}
```

Please consult your editor's documentation.

---

[3]in computer jargon, 'standard ASCII characters' are the characters whose code is included between 32 (space) and 126 (tilde).

7.4 Tools/Hyphenation

Although LaTeX usually does a good job at hyphenating words, sometimes manual intervention may yield better results. Manual hyphens are specified inserting `\-` where we want the word to be broken. A better way is to declare hyphenation rules:

```
\hyphenation{ge-o-phy-sics ge-o-lo-gy earth}
```

The above declaration instructs LaTeX not to hyphen the word 'earth'. Another way to prevent a word to be hyphenated is to put it in `\mbox`:

```
Do not hyphen \mbox{internationalisation}, please. I'm a masochistic.
```

7.5 Tools/Spell Check

LaTeX is not aware of spell spelling. This task is done using external tools like `ispell`, `aspell` or others. Under UNIX, you can use `ispell` this way:

```
shell> ispell -t mydocument.tex
```

The `-t` switch instructs `ispell` to ignore TeX and LaTeX commands. If your language is not English, specify the appropriate dictionary with the `-d` switch:

```
shell> ispell -d italiano -t mydocument.tex
```

8 The **Help** Menu

There are many ways of getting help with LaTeX, both online and offline. The best place to start is the CTAN site, http://www.ctan.org/tex-archive/info/.

- `info latex` (UNIX systems) gives a concise but very complete on-line summary of commands and concepts;

- http://www.giss.nasa.gov/latex/ is a comprehensive on-line reference. Lots of useful links!

- http://www.ctan.org/tex-archive/info/LatexHelpBook/ is a very nice help system for LaTeX, fully integrated with Windows.

- don't forget the news:comp.text.latex newsgroup: it's an invaluable source of help.

As of 2010, most GNU/Linux distributions ship with `TeXLive`, probably the most complete TeX/LaTeX systems. A lot of documentation is provided; on my Ubuntu machine, it's found in `/usr/share/doc/texlive-doc/`.

9 The End

This document is copyleft © Guido Gonzato, 2001–2010, and released under the GNU Free Documentation Licence. I really hope you'll find this guide useful. For any suggestions or comments, please feel free to contact me.

A   Document Templates

A template for the class `article` was presented in Section . More examples are shown in the following figures.

```
\documentclass[twoside,11pt]{book}
\begin{document}
\frontmatter
\begin{titlepage}
\title{The Book of Mine}
\end{titlepage}
\author{John B. Smith}
\maketitle
\tableofcontents
\mainmatter
\part{The Beginning}
\chapter{Introduction}
\section{Let's Start}
The book starts here.
\part{The End}
\backmatter
Thank you for reading this book.
\end{document}
```

Figure 4: Book template.

```
\documentclass[twoside,12pt]{report}
% tables and figures at the end:
\usepackage{endfloat}
\begin{document}
\title{Final Report}
\author{John B. Smith}
\date{London, \today}
\maketitle
\begin{abstract}
This is the final report.
\end{abstract}
\tableofcontents
\listoftables
\listoffigures
\part{Start}
\chapter{Begin}
\section{Introduction}
The report starts here.
\end{document}
```

Figure 5: Report template.

```
\documentclass[12pt]{letter}
\begin{document}
\address{My address}
\signature{Guido}
\begin{letter}{John's address}
\opening{Dear John,}
Thank you for being my friend.
\closing{Hope to see you soon,}
\ps{P.S. Say hello to granny!}
\encl{My son's photographs!}
\end{letter}
\end{document}
```

Figure 6: Letter template.

```
\documentclass[a4paper]{article}
\usepackage{type1cm}
\usepackage{times}
\usepackage{color}
\usepackage{rotating}
\pagestyle{empty}
\begin{document}
\begin{sidewaysfigure}
  \fontsize{2.5cm}{2.5cm}\selectfont
  \centerline{\textcolor{blue}{\textbf{Please:}}}
  \vskip 1cm
  \fontsize{4cm}{3cm}\selectfont
  \centerline{\textcolor{red}{DO NOT}}
  \centerline{\textcolor{red}{SMOKE}}
  \centerline{\textcolor{red}{HERE!}}
  \vskip 1cm
  \fontsize{2cm}{2cm}\selectfont
  \centerline{\textcolor{magenta}{If you do,}}
  \centerline{\textcolor{magenta}{you'll be \emph{deboned!}}}
\end{sidewaysfigure}
\end{document}
```

Figure 7: How to write a notice.

```
\documentclass{article}
\usepackage[absolute,showboxes]{textpos}
\usepackage{color}
\usepackage{framed}
\usepackage{graphicx}
\setlength{\TPHorizModule}{10mm} % standard unit of length
\setlength{\TPVertModule}{\TPHorizModule}
\setlength{\TPboxrulesize}{1pt}  % box line width
% start everything near the top-left corner
\textblockorigin{0mm}{0mm}

\begin{document}
\setlength{\parindent}{0pt}
\definecolor{shadecolor}{rgb}{0.9,1,1}
\begin{textblock}{5}(0,0)
% this block is 5 modules wide; height is
% automatically determined
\begin{center}
  \begin{minipage}[c]{0.8 \linewidth}
  \begin{shaded}
  This block is placed with its top left corner at the 'origin'
  on the page, which has been set to (0mm,0mm). The internal
  margin and the shading are provided by the \texttt{minipage}
  and \texttt{shaded} environments.
  \end{shaded}
  \end{minipage}
\end{center}
\end{textblock}
\begin{textblock}{6}(10,1)
  \includegraphics[width=6cm,angle=-90]{gnuplot.ps}
  This picture is at (10,1). Note that rotating it
  by -90 makes it overflow the margin.
\end{textblock}
\begin{textblock}{5}[0.5,0.5](2.5,8)
This block is at position (2.5,8), but because the optional
argument [0.5,0.5] has been given, it is the centre of the block
which is located at that point, rather than the top-left corner.
\end{textblock}
\begin{textblock}{3,4}(6,4)
The dimensions of this block are 3$\times$4 cm.
Its origin is position (6,4) on the page. Note that the text
overflows the margin in some cases; you'll want to
use the \texttt{minipage} environment to prevent that.
\end{textblock}
\end{document}
```

Figure 8: How to write a poster.