An introduction to typesetting with \LaTeX

Peter Flynn
Copyright

This document is copyright © 1999–2017 by Silmaril Consultants under the terms of the GNU Free Documentation License (copyleft).

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in Appendix 6 starting on page 271.

You are allowed to distribute, reproduce, and modify this document without fee or further requirement for consent subject to the conditions in section F.5 on page 275. The author has asserted his right to be identified as the author of this document. If you make useful modifications you are asked to inform the author so that the master copy can be updated for the benefit of others.

The PDF edition was typeset with XeLaTeX in Charis SIL, Raleway, and Nimbus15 Narrow from the XML master document also used in the web and eBook editions, transformed with XSLT2, Calibre, and kindlegen. Readers of the Kindle (.mobi) version may notice that the \TeX{} and related logos are reproduced in plain text rather than with the special spacing and alignment normally seen; this is to compensate for the many deficiencies of the Kindle file format and the Kindle devices, which are not capable of reproducing microtypographic adjustments. Standard EPUB3 eBook readers are not affected by this and show the logos in their correct form.
Formatting Information

An introduction to typesetting with \LaTeX

Peter Flynn

4th December 2018
Contents

Welcome to Formatting Information ........................................... vii
Credits .................................................................................. ix
Foreword .................................................................................. xi
Preface ..................................................................................... xv
Introduction .............................................................................. xxi

1 Writing documents ............................................................... 1
  1.1 Markup ............................................................................... 2
  1.2 Choosing your \LaTeX{} processor ......................................... 4
  1.3 Quick start for the impatient .............................................. 7
  1.4 \LaTeX{} commands ......................................................... 9
     1.4.1 Simple commands ................................................... 12
     1.4.2 Commands with arguments ....................................... 13
  1.5 White-space in \LaTeX{} .................................................... 13
     1.5.1 Swallowing space after commands ........................... 13
  1.6 Special characters .......................................................... 15
     1.6.1 Using the special characters ...................................... 15
  1.7 Quotation marks ............................................................... 17
  1.8 Accents ............................................................................. 19
  1.9 Dimensions, hyphenation, justification, and breaking .......... 23
     1.9.1 Specifying size units ............................................... 23
     1.9.2 Hyphenation ........................................................... 26
     1.9.3 Breakable and unbreakable text ............................... 27
     1.9.4 Dashes .................................................................... 28
     1.9.5 Justification ............................................................ 29
     1.9.6 Languages ............................................................... 30
  1.10 Mathematics ................................................................. 31

2 Basic structures ...................................................................... 33
  2.1 The Document Class Declaration ...................................... 34
     2.1.1 Document classes ................................................... 34
     2.1.2 Extending the default classes .................................... 35
     2.1.3 Document class options .......................................... 36
  2.2 The document environment ............................................. 39
  2.3 Titling ............................................................................... 42
  2.4 Abstracts and summaries ................................................. 44
  2.5 A little think about structure ............................................ 45
  2.6 Sections ............................................................................ 47
     2.6.1 Section numbering .................................................. 48
  2.7 Ordinary paragraphs ....................................................... 49
  2.8 Table of contents ............................................................ 51
3 Packages and CTAN

3.1 Packages and classes ................................................. 56
  3.1.1 Using an existing class ........................................ 57
  3.1.2 Using an existing package .................................... 57
  3.1.3 Package documentation ....................................... 59
3.2 Installing classes and packages .................................. 59
  3.2.1 Downloading packages ......................................... 60
  3.2.2 Installing a class or package manually ....................... 61
  3.2.3 Replicating the TDS .......................................... 66
3.3 Where to go for help ............................................... 67
  3.3.1 Beginners start here .......................................... 67
  3.3.2 The Minimal [Non-]Working Example or MWE ................. 67
  3.3.3 The FAQ ..................................................... 67
  3.3.4 StackExchange .............................................. 68
  3.3.5 The \TeX{}hax mailing list .................................. 68
  3.3.6 Web sites ................................................... 68
  3.3.7 Usenet News ............................................... 68
  3.3.8 Google \LaTeX{} list ........................................ 69
  3.3.9 Commercial support ......................................... 69
4 Lists, tables, figures .................................................. 71
  4.1 Lists ........................................................................ 72
    4.1.1 Itemized lists .............................................. 73
    4.1.2 Enumerated lists .......................................... 73
    4.1.3 Description lists .......................................... 74
    4.1.4 Inline lists ............................................... 75
    4.1.5 Reference lists and segmented lists ....................... 77
    4.1.6 Lists within lists ........................................ 77
  4.2 Tables ....................................................................... 79
    4.2.1 Floats ...................................................... 79
    4.2.2 Normal tables .............................................. 81
    4.2.3 Simple tabular matter ..................................... 81
    4.2.4 More complex tabular formatting ......................... 85
    4.2.5 More on tabular spacing ................................... 86
    4.2.6 Tabular techniques for alignment ......................... 88
  4.3 Figures ..................................................................... 89
  4.4 Images ..................................................................... 91
    4.4.1 Image file formats ......................................... 91
    4.4.2 Resizing images ............................................ 93
    4.4.3 Making images ............................................. 94
    4.4.4 Graphics storage ........................................... 96
  4.5 Quotations ............................................................ 97
  4.6 Boxes, sidebars, and panels ...................................... 99
    4.6.1 Boxes of text ............................................... 99
    4.6.2 Framed boxes ............................................... 101
CONTENTS

4.6.3 Sidebars and panels .................................................. 102
4.7 Verbatim text ............................................................... 102
  4.7.1 Inline verbatim ....................................................... 102
  4.7.2 Display verbatim .................................................... 105

5 Textual tools ............................................................... 107
  5.1 Footnotes and end-notes ............................................... 107
  5.2 Marginal notes ........................................................... 109
  5.3 References and citations .............................................. 109
    5.3.1 Cross-references .............................................. 110
    5.3.2 Bibliographic references .................................... 111
  5.4 Indexes and glossaries .............................................. 123
    5.4.1 Indexes .......................................................... 123
    5.4.2 Glossaries ...................................................... 124
  5.5 Multiple columns ..................................................... 127

6 Layouts and fonts .......................................................... 129
  6.1 Changing layout ........................................................ 129
    6.1.1 Margins and spacing ........................................... 131
    6.1.2 Headers and footers ............................................ 135
  6.2 Using fonts ............................................................ 137
    6.2.1 Setting up fonts for \texttt{XeLaTeX} ......................... 137
    6.2.2 Changing the default font family with \texttt{XeLaTeX} ... 139
    6.2.3 Changing the default font family with \texttt{LaTeX} and \texttt{PDFLaTeX} ... 141
    6.2.4 Changing the font-family temporarily with \texttt{XeLaTeX} ... 146
    6.2.5 Changing the font-family temporarily with \texttt{LaTeX} and \texttt{PDFLaTeX} ... 147
    6.2.6 Changing type style ........................................... 149
    6.2.7 Font sizes ...................................................... 151
    6.2.8 Logical markup ............................................... 153
    6.2.9 Colour .......................................................... 155
  6.3 The \texttt{LaTeX} font catalogue .................................... 157

7 Programmability ............................................................ 161
  7.1 Simple replacement macros ....................................... 161
  7.2 Macros using information gathered previously .................... 162
  7.3 Macros with arguments ............................................. 165
  7.4 Nested macros ....................................................... 167
  7.5 Macros and environments ......................................... 168
  7.6 Reprogramming \texttt{LaTeX}'s internals .......................... 170
    7.6.1 Changing list item bullets ................................... 171

Formatting Information
FORMATTING INFORMATION

8 Compatibility................................................. 173
  8.1 Converting into \LaTeX ....................................... 175
      8.1.1 Getting \LaTeX out of XML .............................. 177
  8.2 Converting out of \LaTeX ................................... 182
      8.2.1 Conversion to Word ................................... 183
      8.2.2 The \texttt{latex} package .............................. 184
      8.2.3 \LaTeX2HTML .......................................... 184
      8.2.4 \TeX\texttt{4ht} ..................................... 185
      8.2.5 Extraction from PostScript and PDF .................... 185
      8.2.6 Last resort: strip the markup ......................... 186
  8.3 Going beyond \LaTeX ....................................... 186

A Installation.................................................. 189
  A.1 Installing the software .................................... 190
      A.1.1 Unix and GNU/Linux .................................. 190
      A.1.2 Apple Mac OS X ....................................... 193
      A.1.3 Microsoft Windows .................................... 202
  A.2 Your personal \TeX\ directory .............................. 217
      A.2.1 Create a personal \TeX folder in Unix and GNU/Linux .. 217
      A.2.2 Create a personal \TeX folder in Apple Mac OS X .......... 218
      A.2.3 Create a personal \TeX folder in Microsoft Windows .. 218
  A.3 Picking an Editor .......................................... 222
  A.4 Installation problems ...................................... 224
  A.5 Configuring \TeX search paths ............................ 226

B Installing new fonts ......................................... 229
  B.1 TrueType and OpenType fonts ............................... 230
  B.2 Installing \METAFONT fonts ................................ 230
  B.3 Installing PostScript fonts ................................ 231
      B.3.1 Font Definitions ..................................... 239
  B.4 Updating your font maps ................................... 240
      B.4.1 Updating font maps on Mac and Linux (and \TeX Live on Windows) 241
      B.4.2 Updating font maps in \MiKTeX .......................... 241

C Commands, errors, and viewing ............................... 243
  C.1 Terminals and windows ..................................... 244
      C.1.1 So where is the terminal window? ....................... 245
      C.1.2 Using the terminal window ............................ 246
      C.1.3 How do I know what commands to type? ................ 247
  C.2 Typesetting .............................................. 249
  C.3 Errors and warnings ....................................... 249
      C.3.1 Error messages ......................................... 250
      C.3.2 Warnings ............................................... 251
  C.4 Screen preview ............................................ 255
Welcome to Formatting Information

This is the print version of Formatting Information, a book about how to use the \LaTeX{} document preparation system. \LaTeX{} takes over where wordprocessors and desktop publishing systems leave off, making it possible to automate your formatting consistently, accurately, and reusably, without the tedious and repetitive manual formatting required by other systems.

This book has helped thousands of users get started. It's now at version seventh (2018) but this is an interim release: everything has been tested but some details remain to be rewritten — this has now to wait until the next release of \TeX{} Live (2016) towards the end of the year. The only things you need are a computer and a copy of \LaTeX{} (free or commercial)...and a document that you want to typeset. \LaTeX{} works on almost any computer, and you can download it from the TUG web site, install it from the TUG DVD, or buy one of the excellent commercial versions.

In the web and eBook editions, this page doubles as the index, but in the print (PDF) edition, the index is at the end (p. 287). If you haven’t done any typesetting before, I recommend that you start at the beginning. If you’re itching to get started, and you feel you know enough about computers and text-editing already, you can try the Quick Start instead.

Either way, welcome to \LaTeX{}. Take it gently for a while, and get used to being able to spend more time actually writing than formatting. If you find mistakes, please let me know so that I can correct them.

Some font conventions are used in the text and the index to distinguish between different meanings. These are listed in ‘Symbols and conventions’ on page xxix. The entries in the index are all hyperlinked to their source. In the web and eBook editions, subsequent multiple occurrences give the section number or name. Page or section numbers in bold type indicate the location where the entry is explained.
Credits

Earlier editions of ‘Formatting Information’ were prompted by the generous help I received from \TeX users too numerous to mention individually. Shortly after TUGboat published it (Flynn, 2002), I was reminded by a spate of email of the fragility of documentation for any system which is constantly under development. While the core of \LaTeX is as stable as ever, there have been revisions to packages, issues of new distributions, new tools, new interfaces, new books and online documents, corrections to my own errors, suggestions for rewording, and in one or two cases mild abuse for having omitted package X which the someone felt to be indispensable.

The last few editions have been the result of a few years of allowing it to lie fallow, accumulating suggestions and finding errors, but taking on board the large number of changes which daily pass in front of all of us who read comp.text.tex and tex.stackexchange.com, and the sometimes more obvious changes visible when one installs a new version of \TeX. The previous edition came after a longer pause while I finished my research into editing interfaces for structured documents (Flynn, 2014a), which took rather longer than I expected, so there was rather more to change; plus I switched the web version to a HTML5 mobile layout, which meant reprogramming the transformation. The new print editions now use \TeXl, which has meant no more worrying about stray UTF-8 characters, and the ability to use different fonts. This edition now assumes the reader uses \TeXl with biblatex with biber, and all the examples and package references have been updated to match.

I am grateful as always to the people who sent me corrections and suggestions for improvement. Please keep them coming: only this way can this book reflect what people want to learn. The same limitation still applies, however: no mathematics, as there are already a dozen or more excellent books on the market as well as many online documents, some listed in ‘Where’s the math?’ on page xxiv, which deal with mathematical typesetting in \TeX and \LaTeX in finer and better detail than I am capable of.

As I was finishing an earlier edition, I was asked to review an article for ‘The Prac\TeX Journal’, which grew out of the Practical \TeX Conference in 2004. At that meeting, Peter Flom specifically took
the writers of documentation to task for failing to explain things more clearly, and as I read more, I found myself agreeing, and resolving to clear up some specific problems areas as far as possible. I was delighted to see at subsequent Practical \TeX{} Conferences, in 2006 and later, that more presenters, especially in the Humanities, have stepped up to Peter’s challenge.

It is very difficult for people who write technical documentation to remember how they struggled to learn what has now become to them a familiar system. So much of what we do is second nature, and a lot of it actually has nothing to do with the software, but more with the way in which we view and approach information, and with our general level of knowledge of computing. As computer systems become more sophisticated, they require less detailed knowledge from users, even while the takeup of computer usage rises. The result is a generation of users who know what they want, but who are wholly incapable of knowing when they’ve got it, and lack the vocabulary and the experience to explain how to get it; who have only ever seen one way of doing something, and believe that if the result looks pretty, it means it must be right. As technical writers, we need to explain why, not just how, so if I have obscured something by making unreasonable assumptions about your knowledge, please let me know so that I can correct it.
Foreword

This document originally accompanied a two-day introductory training course. It became obvious from repeated questions in class and afterwards, as well as from general queries on comp.text.tex that many, perhaps most, users do not read the FAQs, do not use the TUG web site or the CTAN repositories, do not buy the books and manuals, do not use the newsgroups, mailing lists, or web forums, and do not download or read the excellent free documentation.

Instead, they try to get by, using the time-honoured training technique known as 'sitting by Nellie', which involves looking over a colleague’s shoulder in the office, lab, library, pub, or classroom, and absorbing all of ‘Nellie’’s bad habits along with the good ones. And they use guesswork or imagination for the rest, Googling topics to try and find an answer, but unable to distinguish a reliable site from a poor one.

People do this for many reasons: shortage of time, lack of information (no-one ever told them there was free documentation), dislike of reading manuals, or even just laziness (my own excuse). But chiefest of reasons is that so much of the existing documentation is written for people who are already experts at reading documentation, as well as being expert in using \LaTeX. Most beginners don’t want extensive reasoning over all the available choices: they want simple, direct, prescriptive instruction. If you want one of these, do this.

In the summer of 2001 I presented a short proposal at the annual TUG conference, held that year at the University of Delaware, on the marketing of \LaTeX (Flynn, 2001), and I showed an example of a draft brochure designed to persuade newcomers to try \LaTeX for their typesetting requirements. As a result of questions and suggestions, it was obvious that it needed to include a pointer to some documentation, and I agreed to make available a revised form of the document you are now reading, expanded to be used outside the classroom, and to include those topics on which I have had most questions from users over the years.

It turned out to mean a significant reworking of a lot of the material. Some of it appears in almost every other manual and book on \LaTeX but it is essential to the beginner and therefore bears repetition. Some of it appears in other forms elsewhere, and is included here because I felt
Foreword

it needed explaining. And some of it appears nowhere else but this document. I took the opportunity to revise the structure of the training course in parallel with the book, and to include a more comprehensive index. It is by no means perfect, and I would be grateful for comments and corrections to be sent to me at the address given on the home page (at the front of the book, if you are reading this on paper or in an ebook). As I also noted earlier, it can be used as a one-day course if the users already have some experience of writing, editing, or reviewing formal documents for publication or assessment (eg reports, white papers, essays, books, theses, articles, etc).

I had originally hoped that the LaTeX version of the document would be processable by any freshly-installed default LaTeX system, but the need to include font samples which go well beyond the default installation, and the need to use some less-common packages which the new user is unlikely to have installed, meant that this document itself was not really a simple piece of LaTeX, no matter how simply it may describe the process itself. That was then; nowadays I would hope most people install the full works, so everything should work as-is.

However, as the careful reader may already have noticed, the master source of this document is not maintained in LaTeX but in XML. Installations of TeX are becoming more comprehensive, which means that modern systems are likely to include all the fonts and packages needed, so what I called last time ‘a future task’ is now creeping up fast: to rewrite the XSLT transformation to that it can be guaranteed to process with all of the current full LaTeX installations.

If you are just starting with LaTeX, at an early opportunity you should buy or borrow a copy of *LaTeX: A Document Preparation System* which is Lamport’s original manual. More advanced users should get the *Companion*, the *Graphics Companion* and the *Web Companion*. Mathematical users might want to start with the *Short Math Guide*. Details are in section F.13 on page 283.
Published resources

There are hundreds if not thousands of web pages about how to use \LaTeX{}, and the online version of this book is just one. There are dozens of books about \LaTeX{} too, and these are a few of the printed ones that I strongly recommend:

- van Dongen’s \textit{\LaTeX{} and Friends} covers \LaTeX{} and mathematics with reference to the latest packages and techniques. I’ve known Marc for many years: he works at the same institution that I do, and this is a shameless plug for a really excellent book.

- Oetiker’s \textit{A (Not So) Short Introduction to \LaTeX{}2e} started as a translation and rationalisation of a ground-breaking German-language introduction to \LaTeX{}. It has since taken on a momentum of its own, and has itself been translated into many languages. It remains the best short guide, even if it has grown a bit.

- Grätzer’s \textit{Practical \LaTeX{}} is a very approachable, non-technical introduction to everyday \LaTeX{} usage. The first chapter is designed as a standalone guide to get you started.

There is a much bigger list of online resources on the \TeX{} Users Group \texttt{web site}, and a much longer list of books there at \texttt{tug.org/books/}.

If you’ve just finished a \LaTeX{} course, or read a book or web site, and you want a refresher or aide-mémoire, my leaflet ‘The Very Short Guide to \LaTeX{}’ is what it says it is: a 4-page guide to the bare essentials of document construction, designed as a reminder of what you’ve just learned.
Preface

Many people discover \LaTeX{} after years of struggling with wordprocessors and desktop publishing systems, and are amazed to find that \TeX{} has been around for over 30 years and they hadn’t heard of it. It’s not a conspiracy, just ‘a well-kept secret known only to a few million people’, as one user has put it.

Perhaps a key to why it has remained so popular is that it removes the need to fiddle with the formatting while you write. Playing around with fonts and formatting is highly attractive, not just to new computer users, and it’s great fun, but it is completely counter-productive for the serious author or editor who needs to concentrate on actual writing — ask any journalist or professional writer. ‘Best-guess’ estimates by experts in the field of usability engineering are that average computer users may spend up to 50% of their time fiddling with the formatting rather than thinking or writing — and this is with the so-called ‘office productivity software’ that the major manufacturers foist on their clients!

A few years ago a new \LaTeX{} user expressed concern on the \texttt{comp.text.tex} newsgroup about ‘learning to write in \LaTeX{}’. Some excellent advice was posted in response to this query, which I reproduce with permission below (the bold text is my own emphasis):

No, the harder part might be writing, period. TeX/LaTeX is actually easy, once you relax and stop worrying about appearance as a be-all-and-end-all. Many people have become ‘Word Processing Junkies’ and no longer ‘write’ documents, they ‘draw’ them, almost at the same level as a pre-literate 3-year-old child might pretend to ‘write’ a story, but is just creating a sequence of pictures with a pad of paper and box of Crayolas — this is perfectly normal and healthy in a 3-year old child who is being creative, but is of questionable usefulness for, say, a grad student writing a Master’s or PhD thesis or a business person writing a white paper, etc. For this reason, I strongly recommend not using any sort of fancy GUI ‘crutch’. Use a plain vanilla text editor and treat
it like an old-fashioned typewriter. Don’t waste time playing with your mouse. Note: I am not saying that you should have no concerns about the appearance of your document, just that you should write the document (completely) first and tweak the appearance later...not [spend time on] lots of random editing in the bulk of the document itself. (Heller, 2003)

More recently, an article reporting on a study of writing patterns between Microsoft Word users and \LaTeX{} users reported that it was faster to use Word (Knauff & Nejasmic, 2014). As a reviewer of that article, I asked the authors to make it clearer that the use of the proper templates (classes and packages) removed the need for \LaTeX{} users to spend the time formatting that Word users do. The publication of the article upset a number of people in the \LaTeX{} field, but I hope that it will spur the critical examination of how we write, and why it’s better to do it in \LaTeX{} than in other systems.

Learning to write well can be hard, but authors shouldn’t have to make things even harder for themselves by using manually-driven systems which break their concentration every few seconds for some footling adjustment to the appearance, simply because the software is incapable of doing it right by itself.

Donald Knuth originally wrote \TeX{} to typeset mathematics for the second edition of his master-work *The Art of Computer Programming* (Knuth, 1980), and it remains pretty much the only typesetting program to include fully-automated mathematical formatting by default, done the way mathematicians do it. But he also brought out a booklet called *Mathematical Writing* (Knuth, Larrabee & Roberts, 1989) which shows how important it is to think about what you write, and how the computer should be able to help, not hinder, the author while writing.

But \TeX{} is much more than math: it’s a programmable typesetting system which can be used for almost any formatting task, and the \LaTeX{} document preparation system which is built on \TeX{} has made it usable by almost anyone. Professor Knuth generously placed the entire \TeX{} system in the public domain, which meant it is free for anyone to use, but for many years this also meant that there was little commercial publicity which would have got \TeX{} noticed outside the technical field,
because there was no great corporate marketing department to advertise its existence. Even now, some people who used it in college believe that it no longer exists!

Nowadays, however, there are several companies selling \TeX{} software or services,\textsuperscript{1} dozens of publishers accepting \LaTeX{} documents for publication, and hundreds of thousands of users using \LaTeX{} for millions of documents.\textsuperscript{2}

There is occasionally some confusion among newcomers between the two programs, \TeX{} and \LaTeX{}, and the other versions available, so I’d like to clear this up:

\TeX{} : The underlying typesetting program, originally written by Donald Knuth at Stanford in 1978–79. It implements a macro-driven typesetters’ programming language of some 300 basic operations, and it has formed the core of many other desktop publishing (DTP) systems. Although it is still possible to write in the raw \TeX{} language, you need to study it in depth, and you need to be able to write macros (subprograms) to perform even the simplest of repetitive tasks.

\LaTeX{} : A user interface for \TeX{}, designed by Leslie Lamport while at Digital Equipment Corporation (DEC) in 1985 to automate the common tasks of document preparation. It provides a simple way for authors and typesetters to use the power of \TeX{} without having to learn the underlying language. \LaTeX{} is the recommended system for all users except professional typographic programmers and computer scientists who want to study the internals of \TeX{}.

\ConTeXt{} : (not ‘Contest’!) A system similar to \LaTeX{}, but with its own set of commands, and a much greater emphasis on producing high-function PDF output. The documentation is less accessible than for \LaTeX{}, but the author, Hans Hagen, provides excellent support at Pragma/ADE.

\PDF\TeX{} and \PDFL\a\TeX{} : Extended versions of \TeX{} and \LaTeX{} that create PDF instead of DVI files, written by Hân Thế Thành. There are

\textsuperscript{1} See, for example, the list of \TeX{} vendors in Table 1 on page xxix, and the list of consultants published by TUG.

\textsuperscript{2} A guesstimate. With free software it’s virtually impossible to tell how many people are using it.
Debunking the mythology

Naturally, over all the years, a few myths have grown up around \LaTeX, often propagated by people who should know better. These *canards* make it harder to explain to potential users why they should look at \LaTeX, so, just to clear up any potential misunderstandings.

**MYTH: \LaTeX has only got one font**: \LaTeX systems can use any OpenType, TrueType, Adobe (PostScript) Type1 or Type3 (METAFONT) font. This is more than any other known typesetting system. \LaTeX’s default font is Computer Modern (based on Monotype Series 8; see Table 6.2 on page 142), not Times Roman, and some people get very upset because Computer Modern looks different to Times (I’m not making this up: it’s just a typeface, people, get over it).

**MYTH: \LaTeX isn’t WYSIWYG**: Simply not true. \TeX’s DVI and PDF is generally better quality WYSIWYG than any wordprocessor and most DTP systems. What people mean is that the typographic display (preview) is asynchronous with the editor window. This is only true for the default CLI implementations. See the list item ‘Synchronous typographic displays’ on page xxviii for details of synchronous versions.

**MYTH: \LaTeX is obsolete**: Quite the opposite: it’s under constant development, with new features being added or updated almost daily. Check comp.text.tex for messages about recent uploads to CTAN. It’s arguably more up-to-date than most other systems: \LaTeX had the Euro (€) before anyone else, it had Inuktitut typesetting before the Inuit got their own province in Canada, and it still produces better mathematics than anything else.

also enhancements for micro-typographic extensions, native font embedding, and PDF support for hyperlinking. It is currently (2016) still the default \TeX engine in most distributions.

**Xe\TeX and Xe\LaTeX**: A recent reimplementation of \TeX by Jonathan Kew which merges Unicode and modern font technologies. It is in common use in editing environments such as \TeXshop (Apple Macintosh OS X), *Kile* (Unix & GNU/Linux), and Win\Edt
More mythology

If you come across other myths from people who should know better, please let me know — I’m collecting them here!

**MYTH: ‘\LaTeX{} is a Unix system’:** People are also heard saying it’s ‘a Windows system,’ ‘a Mac system,’ etc. etc \textit{ad nauseam}. \LaTeX{} systems run on almost every computer in use, from the biggest supercomputers right down to handhelds (even old PDAs like the Sharp Zaurus and the Nokia N800), and most Apple and Android smartphones). That includes Unix & GNU/Linux, including Apple Macintosh OS X, Windows, and all other desktop, mini, and mainframe systems. If you’re using something \LaTeX{} doesn’t run on, it must be either incredibly new, incredibly old, or unbelievably obscure.

**MYTH: ‘\LaTeX{} is “too difficult”’:** This has been heard from physicists who can split atoms; from mathematicians who can explain why \(\pi\) exists; from business people who can read a balance sheet; from historians who can explain Byzantine politics; from librarians who can understand LoC and MARC; and from linguists who can decode Linear ‘B’. It’s complete nonsense: most people can grasp \LaTeX{} in 20 minutes or so — it’s not rocket science (or if it is, I know any number of unemployed rocket scientists who will teach it to you).

**MYTH: ‘\LaTeX{} is “only for scientists and mathematicians”’:** Completely untrue. Although \TeX{} grew up in the mathematical and computer science fields, because those were its author’s fields, two of its biggest growth areas are in the humanities and business, especially since the rise of XML brought new demands for automated web-based typesetting.

(Windows). Details are at the Sourceforge web site. Xe\LaTeX{} is used to produce the PDF edition of this book.

**\TeX{}info:** \TeX{}info is the official documentation format of the GNU project.\(^3\) It was invented by Richard Stallman and Bob Chassell. It

\(^3\)GNU’s Not Unix (GNU) is a project to create a completely free computing system — ‘free’ meaning both free from encumbrances and restrictions as well as free of charge.
uses a single source file to produce output in a number of formats, both online and printed (DVI, HTML, INFO, PDF, XML, etc). TeXinfo documents can be processed with any \TeX\ engine.

Both \TeX\ and \LaTeX\ have been constantly updated since their inception. Knuth has now frozen changes to the \TeX\ engine so that users and developers can have a bug-free, rock-stable platform to work with.\footnote{Knuth still fixes bugs, although the chances of finding a bug in \TeX\ these days approaches zero.} Typographic programming development continues with the New Typesetting System (NTS), planned as a successor to \TeX. The \LaTeX\X\ project has taken over development of \LaTeX, and the current version is \LaTeX\X{}2e, which is what we are concentrating on here. Details of all developments can be had from the TUG web site at \url{www.tug.org}
Introduction

This book originally accompanied a two-day course on using the \LaTeX\ typesetting system. It was extensively revised and updated for publication, so that it could be used for self-study as well as in the classroom. For those with sufficient prior knowledge of computing and authoring, it has also successfully been used as the basis for a 1-day intensive introductory course. It is aimed at users of Unix & GNU/Linux, including Apple Macintosh OS X, and Microsoft Windows systems, but it can be used with \LaTeX\ on any platform, including other Unix workstations, mainframes, Android and Apple smartphones, and even some older Personal Digital Assistant (PDA)s.

Who needs this book?

The course was originally designed for computer-literate but non-IT professionals in business, academic, and nonprofit organisations. You may be in a similar position, but you may also come from another background entirely; you may be a hobbyist, a school or college student, a home computer user or a volunteer worker, or you might just be interested in high-quality automated typesetting. However, it’s likely that you have one or more of the following or similar objectives:

- producing consistent, typeset-quality formatting;
- formatting long or complex or highly-structured or repetitive or automatically-generated documents;
- saving time and effort by automating common tasks;
- gaining independence from expensive and restrictive proprietary hardware, software, or file formats;
- creating robust, durable documents which will survive changes in technology.

Skills needed

\LaTeX\ is a very easy system to learn, and requires no specialist knowledge to get started, although it’s useful if you understand something about writing, formatting, and readability. However, you do need to be
completely familiar with using your computer, which means knowing the following topics thoroughly. Note that none of these is in any way specialist; they’re all basic, fundamental, standard computer skills that everyone should know:

- **Using the mouse**: how to point and click with your mouse to select text and pick from a menu (or to use keyboard shortcuts to do the same)
- **Handling files**: how to create, open, save, close, rename, copy, move, and delete files and folders (directories);
- **Handling characters**: where to find all 95 of the printable ASCII characters on your keyboard and what they mean, and how to type accents and symbols, if you need them;
- **Using an editor**: how to use a good plaintext editor\(^5\);
- **Downloading files**: how to use your Web browser and/or file transfer program to download and save files from the Internet;
- **Unzip files**: how to uncompress and unwrap (unzip or detar) compressed files;
- **Install software**: how to install software, both manually and using automated installers;
- **RTFM**: how to read and follow instructions and how (and where) to ask for help.

If you don’t know how to do these things yet, it’s important to go and learn them first. Trying to become familiar with basic computer skills at the same time as learning \LaTeX{} is not going to be as effective as doing them in the right order.

It is important to understand that these are not specialist skills — they are standard for anyone who uses a computer, and they form a fundamental part of the basic knowledge of computers. With the exception of software installation, they were included in the European Computer Driving Licence (ECDL) course: the relevant module and

\[^5\] Not a wordprocessor like OpenOffice, Lotus Notes, Corel WordPerfect, or Microsoft Word, and not a ‘dumb’ editor like Apple TextEdit or Microsoft Notepad.
section numbers of the ECDL syllabus are noted in parentheses or in the margin above (Kelly & O’Connor, 2005).

Objectives of this book

By the end of this book, you should be able to undertake the following tasks:

1. use your editor to create and maintain your documents;
2. use \LaTeX\ markup to identify your document structure and formatting requirements;
3. typeset \LaTeX\ documents, correct simple formatting errors, and display or print the results;
4. identify and use additional \LaTeX\ packages (using the Internet for downloading where necessary and installing them);
5. recognise the limitations of procedural markup systems and choose appropriate generic markup methods where appropriate.

Synopsis

The original course covered the following topics as separate sessions. Earlier versions of this document kept to this structure in the book as chapters, but the current version has moved Installation (originally chapter 1) to Appendix A and merged it with the details of configuration; and Typesetting, viewing, and printing (originally chapter 4) to a new Appendix; as the procedure in both cases has been so much simplified that the previous level of detail is no longer needed in most cases.

1. How to create \LaTeX\ documents (with a Quick-Start Guide for the impatient);
2. Basic document structures (the Document Class Declaration and its layout options; the \texttt{document} environment with sections and paragraphs);
3. Using packages and \texttt{CTAN} to adapt formatting to your needs;
4. Other document structures (lists, tables, figures, images, and verbatim text);

5. Textual tools (footnotes, marginal notes, cross-references, indexes and glossaries, and bibliographic citations);

6. Typographic considerations (white-space and typefaces; inline markup and font changes; extra font installation and automation);

7. Programmability and automation (macros and modifying LaTeX’s behaviour);

8. Conversion and compatibility with other systems (XML, Word, etc).

A Where to get and how to install LaTeX (using the TeX Users Group’s TeX Collection DVD);

B How to install new fonts;

C Typesetting, viewing, and printing (largely obsolete now that editors are better integrated with viewers and printers);

D User groups and the benefits of membership;

E The ASCII character set;

F The GNU Free Documentation License.

I have made a few other changes in the transition to printed and online form, but the basic structure is the same, and the document functions as a workbook for the course as well as a standalone self-teaching guide.

Where’s the math?

Please understand that this document does not cover mathematical typesetting, complex tabular material, the design of large-scale macros and document classes, or the finer points of typography or typographic design, although it does refer to these topics in passing on a few occasions.

There are several other guides, introductions, and ‘get-started’ documents on the Web and on CTAN which cover these topics and more in great detail. Among the more popular are:
Availability of \LaTeX{} systems

\begin{itemize}
  \item \textit{Getting Started}, where all beginners should start;
  \item \textit{A (Not So) Short Introduction to }\LaTeX{}\textit{2e} is a good beginner’s tutorial;
  \item \textit{Gentle Intro} is a classic tutorial on Plain \TeX{} (not \LaTeX{});
  \item \textit{Imported graphics} shows you how to do (almost) anything with graphics: side-by-side, rotated, etc;
  \item \textit{Short Math Guide} gets you started with the American Math Society’s extensions;
  \item \textit{The \LaTeX{} Symbol List} shows over 2,500 symbols available.
\end{itemize}

This list was taken from the CTAN search page. There are also lots of books published about \TeX{} and \LaTeX{}: the most important of these for users of this document are listed at the end of the on page xi.

Availability of \LaTeX{} systems

The standard implementations of \TeX{} and related systems are in the \TeX{} Collection, published annually on DVD by the \TeX{} Users Group. These are all derived from Knuth’s master versions, and adapted for all major platforms (Unix & GNU/Linux, including Apple Mac OS X; and Microsoft Windows). The DVD is sent free to all TUG members and can be obtained from your local user group. You can also download the ISO image file from CTAN to burn your own copy.

Commercial implementations are listed in ‘Commercial implementations’ on page xxviii.

Systems included on the \TeX{} Collection DVD

\textbf{Pro\TeX{}t (Windows)}: This is the popular \textit{MiK\TeX} implementation plus the \textit{\TeX{}Studio} editor.

\textbf{Mac\TeX{} (OS X)}: This is \TeX{} Live plus the \textit{\TeX{}shop} editor (the Mac’s built-in \textit{Preview} is used for the \textit{WYSIWYG} display).

\TeX{} Live (Unix & GNU/Linux, including Apple Macintosh OS X, and Windows): Generic \TeX{} Live for systems without a built-in package distribution.

\textit{Formatting Information}
Users of Red Hat/CentOS, Debian/Ubuntu, and derivative systems should normally use their system-supplied package manager (e.g., `yum`, `rpm`, or `apt-get`; or `Synaptic` or `Software Installer` etc) to install the RPM or DEB packaged versions provided in their supplier’s repositories, as these provide tighter integration with other software such as editors and viewers; but the generic `TeX Live` can be used where the absolute latest bleeding-edge features are required (in which case any system-installed version SHOULD be removed first to avoid conflicts and unresolved dependencies — see the notes in Appendix 1 starting on page 189).

Because the `TeX` program (the internal ‘engine’ which does the actual typesetting) is independent of any other software, it doesn’t have its own editor like a wordprocessor does. Instead, you get to choose whichever editor you prefer: there are lots available, and you can switch between them to find one you like: see ‘Graphical interfaces (editors)’ and section A.3 on page 222 for details.

### Graphical interfaces (editors)

Most users run `TeX` with a graphical plaintext editor which has a toolbar and menus like other windowing applications. These usually include all the common formatting features of `TeX` plus writing tools like spellchecking, thesaurus, indexing, and bibliographic citation, and generally all work in a very similar way. Text-only interfaces are available for use on servers and automated production systems (see ‘Command-line interfaces’ on page xxvii).

The Windows and Mac systems described in ‘Availability of `TeX` systems’ on page xxv come with a recommended editor (`TeXStudio` and `TeXshop` respectively), but you can install any other suitable editor you prefer (see section A.3 on page 222). The Unix & GNU/Linux distribution does not install any editor because these systems usually have their own software repositories with suitable editors already available for installation, such as `Emacs`, `vi`, `TeXStudio`, or `Kile`.

Fully synchronous typographic interfaces (editors) are available as commercial products: see the list item ‘Synchronous typographic displays’ on page xxviii.
Command-line interfaces

While you would use a graphical interface to set up an automated system like a web server or e-commerce environment, it is useless where systems have to run in the background, unattended, with no human to click on buttons. In fact, the \TeX\ typesetting engine is a Command-Line Interface (CLI) program, which can be used from any script or console or ‘Command’ window. You can type the command $\texttt{xelatex}$ followed by the name of your document file (see Figure C.2 on page 246 for an example).

Commands like these let you run \LaTeX\ in an automated or scripted environment like a Common Gateway Interface (CGI) script on a web server or a batch file on a document publishing system. All the popular distributions for all systems, both free and commercial, include this CLI interface as standard.

WYSIWYG displays

\LaTeX\ usually displays your typeset results in a separate window such as a PDF viewer, updated automatically every time the document is retypeset, because the typesetting is kept separate from the editing. This is called an ‘asynchronous’ display. Some systems, however, can format the typesetting while you type each character, like a wordprocessor, although at the expense of some flexibility. These are called ‘synchronous’ displays.

Asynchronous typographic displays: The WYSIWYG display is updated when the document is reprocessed, rather than while you are still typing, as it would with a wordprocessor. To update the display, just click on the button which reformats the document. You are probably already familiar with this idea if you have used HTML, where you reload the page in a browser to see it, or if you have used a spreadsheet, where the ReCalc button (F9) does something similar.

\TeX\ systems typeset the whole document at one go, including all indexing, cross-references, tables of contents, bibliographic citations, and the placement of figures and tables. \TeX\ also formats whole paragraphs at a time, rather than line-by-line as wordprocessors do, in order to get the quality of spacing,
Introduction

hyphenation, and justification right. This approach makes it much faster than a wordprocessor in dealing with typical complex documents, as it can be done without holding the whole document in memory.

**Synchronous typographic displays**: The WYSIWYG display is the editing window, and it updates while you type, like a wordprocessor. Some popular examples are \LaTeX{} (all platforms), *Textures* (Mac), *BaKoMa \TeX{}* (Windows), and *Scientific Word* (Windows) (see Table 1 on page xxix).

With a synchronous display you get Instant Textual Gratification™, but like a wordprocessor, your level of control is restricted to that of the system you use, which cannot provide access to everything that \LaTeX{} can do. For complete control of complex material you may still need to use separate editing and display windows as for asynchronous implementations.

**Near-synchronous displays**: There are a few systems for very-close-to-synchronous WYSIWYG display. These include Jonathan Fine’s *Instant Preview* with the \TeX{} daemon, and David Kastrup’s *preview-latex* package for embedding typographic fragments from the typeset display back into the editor window.

What You See Is What You Get (**WYSIWYG**) refers to the accuracy of the typographical display. Most modern ones are pretty good, given the fact that your screen is probably only a fraction of the accuracy of your printer — between 96 dots per inch (**DPI**) on an old desktop screen and around 300 **DPI** on some handhelds; as opposed to 600 **DPI** on your printer, or 1200 **DPI** or more in photo-quality, and 3,600 **DPI** or more on laser-driven phototypesetters.

**Commercial implementations**

Although the \TeX{} Collection is available free of charge, there are several excellent commercial implementations of \TeX{} and \LaTeX{} listed in Table 1 on page xxix, with enhanced support and additional features. These companies, founders, and staff have been good friends of the \TeX{} and
Table 1: Popular commercial distributions of \TeX systems

<table>
<thead>
<tr>
<th>Product</th>
<th>Platform</th>
<th>Company</th>
<th>URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCTeX</td>
<td>MS-Win</td>
<td>Personal TeX, Inc</td>
<td><a href="http://www.pctex.com/">www.pctex.com/</a></td>
</tr>
<tr>
<td>BaKoMa TeX</td>
<td>MS-Win</td>
<td>Basil K Malyshev</td>
<td><a href="http://www.bakoma-tex.com/">www.bakoma-tex.com/</a></td>
</tr>
<tr>
<td>TrueTeX</td>
<td>MS-Win</td>
<td>True TeX</td>
<td><a href="http://www.truetex.com/">www.truetex.com/</a></td>
</tr>
<tr>
<td>Textures</td>
<td>Mac</td>
<td>Blue Sky Research</td>
<td><a href="http://www.bluesky.com/">www.bluesky.com/</a></td>
</tr>
<tr>
<td>Scientific Word</td>
<td>MS-Win</td>
<td>Mackichan Software</td>
<td><a href="http://www.mackichan.com/">www.mackichan.com/</a></td>
</tr>
<tr>
<td>VTeX</td>
<td>MS-Win, Linux, OS/2</td>
<td>MicroPress, Inc</td>
<td><a href="http://www.micropress-inc.com/">www.micropress-inc.com/</a></td>
</tr>
</tbody>
</table>

\TeX communities for many years: if these versions are of benefit to you, I urge you to support them and buy their products.\(^6\)

**Symbols and conventions**

There are several typographic conventions about how you represent computer-related material in print which are shown in Table 2 on page xxx. Typed commands, keywords, examples of input, and related text are in a fixed-width (monospace) font, like a typewriter, because that’s how program code is usually displayed and edited (this also helps avoid ambiguities, as explained in section 4.7.1.1 on page 103). Special values, like numeric quantities represented by a name or symbol, are in italics, as in mathematics. Terms or references to products, programs, packages, and other components of \TeX have their own typographic form. Finally there are some symbols like keyboard keys and menus, which are shown graphically.

\(^6\) Y&Y, Inc, who produced a \TeX distribution for many years, have ceased trading. Some of their add-on fonts are now being distributed by the \TeX Users Group (see Appendix 4 starting on page 261), or have been replaced by Open Source implementations, and there is a mailing list at the TUG web site for the support of former Y&Y users.


Introduction

Table 2: Typographic notations used in this document

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL</td>
<td>These keywords have a specific and possibly mandatory meaning when shown in THIS STYLE (usually small capitals), according to the specification in RFC 2119 (Bradner, 1997).</td>
</tr>
<tr>
<td>\command</td>
<td>\LaTeX commands (control sequences) that you type which perform an action like \clearpage, or identify your text like \footnote{...}</td>
</tr>
<tr>
<td>\length</td>
<td>Control sequences which store a dimension (a measurement in units), like \textwidth</td>
</tr>
<tr>
<td>counter</td>
<td>Values used for counting (whole numbers only), like section</td>
</tr>
<tr>
<td>term</td>
<td>The defining instance of a new technical term</td>
</tr>
<tr>
<td>environment</td>
<td>A \LaTeX formatting or identification environment, like quotation</td>
</tr>
<tr>
<td>class</td>
<td>A \LaTeX document class (available from CTAN), like memoir</td>
</tr>
<tr>
<td>package</td>
<td>A \LaTeX add-on package (available from CTAN), like footmisc</td>
</tr>
<tr>
<td>product</td>
<td>A program or product name</td>
</tr>
<tr>
<td>typewriter type value</td>
<td>Literal examples</td>
</tr>
<tr>
<td>\author{your name}</td>
<td>Mnemonic examples of things you must type, where you have to supply real-life values of your own, like \author{your name} means you must replace your name with your own real name.</td>
</tr>
<tr>
<td>x</td>
<td>A specific key on your keyboard</td>
</tr>
<tr>
<td>Ctrl + x</td>
<td>Two keys pressed together, not separately</td>
</tr>
<tr>
<td>Esc + q</td>
<td>Two keys pressed one after another</td>
</tr>
<tr>
<td>Submit</td>
<td>An on-screen button to click</td>
</tr>
<tr>
<td>Menu \to Submenu \to Item</td>
<td>A drop-down menu with items to select</td>
</tr>
<tr>
<td>\</td>
<td>A normal space, just so it’s visible</td>
</tr>
</tbody>
</table>

Formatting Information
Production note

This document is written and maintained in XML, using a customized version of the DocBook 5 DTD. Conversions are made to HTML and \TeX using XSLT1.2 scripts and Michael Kay’s Saxon processor.

The complete source, with all ancillary files, is available online at www.ctan.org/tex-archive/info/beginlatex/src/. If you want to try processing it yourself you will need Java and Saxon in addition to a full installation of \TeX.

This document is published under the terms and conditions of the GNU Free Documentation License. Details are in Appendix 6 starting on page 271.
LaTeX documents are all *plaintext* files. This means printable characters only (in whatever writing system is native to your language and culture), no hidden internal binary gubbins like fonts or formatting (except for spaces and linebreaks). If you haven't seen a plaintext file before, it looks like this:

This means printable characters only (in whatever writing system is native to your language and culture), no hidden internal binary gubbins like fonts or formatting (except for spaces and linebreaks).

By comparison, wordprocessor files saying the same thing often actually look something like this inside:

The big advantage of plaintext is not just that it's readable; it's that the files can be copied, downloaded, or uploaded to any computer system running LaTeX and they will typeset exactly the same. Because they are plain text they cannot corrupt your system, and they cannot be used
for hiding virus infections in the way that binary (coded non-plaintext) files can be. Everything you can see is in the file and everything in the file is there for you to see: there is nothing hidden or secret and there are no manufacturers’ proprietary ‘gotchas’ like suddenly going out of date with a new version or imposing selective Digital Restrictions Management (DRM), leaving you unable to open your files.

So, you may ask, if \LaTeX\ files are all plaintext, how does \LaTeX\ know how to format them? The answer is that it uses \textit{markup}: a system of labels which identifies what’s what in your document. \LaTeX\ and its packages recognise the labels and know how to format them, so you don’t usually need to add formatting by hand unless you want to do something very special or invent something out of the ordinary.

\section{1.1 Markup}

In a \LaTeX\ document, you type your text along with \textit{markup} to identify the important bits by name, for example ‘title’, ‘author’, ‘chapter’, ‘section’, ‘figure’, etc. \LaTeX\ does all the typesetting for you automatically, using the markup to apply the formatting rules (styles) you tell it to use.

In the panel ‘Markup’ on p. 3 you can see some examples of markup over the years. In the manuscript era (before printing) scribes often added extra information to what they were copying. In the days of the typewriter, publishers would add markup to the author’s typescript so the printer would know what to do with it. When computers started to be used for text processing, systems tended to follow the established conventions: the similarity between the computer forms is striking, and not coincidental.

\LaTeX\ markup is all in (American) English, with a few abbreviations for long words to minimise typing. Most people use an editor with a menu or toolbar button which knows about markup, so actually typing it is rare except for the beginner.

You do not need to format any of your text \textit{in your editor}, because \LaTeX\ does the formatting all by itself when it typesets. You can of course regularise or neaten its appearance \textit{in your editor} for your own ease of editing (for example, keeping each item in a list on a separate line), but this is not required.
1.1. MARKUP

This term came from printing and publishing, where described the notes on layout or the corrections to make during editing, but the practice of adding annotations to documents goes back to the beginning of writing.

It now means instructions or descriptions added to a computer document to act as guidelines for identification or formatting. Markup has been around for a very long time.

Some of the history of computer markup can be seen in the names (h1 and H1, section, sec, sect1, etc). Anglo-American influence in computing means most common systems are based on English-language names, although they can be used in any language.
CHAPTER 1. WRITING DOCUMENTS

You will often hear \LaTeX markup referred to as ‘commands’ or sometimes ‘control sequences’ (the proper \TeXnical term for them). For all practical purposes these terms all mean the same thing.

1.2 Choosing your \LaTeX processor

Before you go any further there is one configuration you might want to change. As you may have seen in the list on page xvii, there isn’t just one flavour of \LaTeX:

\textbf{Plain \LaTeX}: For many years, there was only \LaTeX, which (like \TeX) produced a .dvi (Device-Independent) file, which had to be converted to Postscript or PDF in an additional step;

\textbf{PDFLaTeX}: In the 1990s, Hàn Thế Thành developed PDF\LaTeX, which (along with PDF\TeX) produced PDF directly, as well as adding benefits like microtypographic adjustments;

\textbf{XeLaTeX}: More recently, Jonathan Kew developed Xe\LaTeX, which not only recognises UTF-8 characters directly, but can also use your system’s natively-installed fonts as well as those which come with \LaTeX.

In this book I am going to recommend that you use Xe\LaTeX if your editor can be set to use it, unless you have a compelling reason not to. In my view the ability to handle natively-installed system fonts as well as UTF-8 characters sets it well above the other processors.

However, there are still some good reasons not to. These include:

- a few (decreasingly few) packages which positively require a processor which creates a .dvi file;

- some specific packages rely on raw Postscript features which need DVI-to-Postscript conversion first, before the PS output can be converted to PDF;

- toolchains which depend on .dvi files;

- not all editors yet make it easy to select Xe\LaTeX as the processor\footnote{Formatting Information}
1.2. CHOOSING YOUR LATEX PROCESSOR

There is a separate but related question of choosing a bibliographic formatter (old-style BibTeX .bst files or the more recent biblatex package) and which bibliographic processor to choose (bibtex or biber). If your documents don’t use bibliographic references, this will not be a concern for you.

The relationship is that the biblatex package and the biber program, like XeLaTeX, deal natively with UTF-8 characters, whereas the .bst files and the bibtex processor have known problems with multibyte (accented and non-Latin) characters, making the reference and citation of works in many languages difficult, if not impossible. We will be dealing with this choice in more detail in section 5.3.2.1 on page 112.

\footnote{Some which are straightforward are shown in Figure 1.1 on the following page.}
CHAPTER 1. WRITING DOCUMENTS

Figure 1.1: Some \LaTeX{} editors being configured to use \XeLaTeX

- *Kile* (Linux, Mac, Windows)
- *TeXnicCenter* (Linux, Mac, Windows)
- *TeXShop* (Linux, Mac, Windows)
- *TeXStudio* (Linux, Mac, Windows)
1.3 Quick start for the impatient

If you already know all about editors and plaintext files and markup and how to run programs, and you know that \LaTeX{} is already fully installed (including an editor that you know how to use), you’d probably like to type something in and see \LaTeX{} do its job.

If you don’t know this stuff yet, then by all means do this section now, but treat it as part of the learning experience. Otherwise you might want to skip forward to section 1.4 on page 9 and read more about how \LaTeX{} works, and come back to this section later.

Figure 1.2: Quick-start example document text

\begin{verbatim}\documentclass[12pt]{article} \usepackage{palatino,url} \setcounter{secnumdepth}{0} \begin{document} \section{My first document} This is a short example of a \LaTeX\ document I wrote on \today. It shows a few simple features of automated typesetting, including: \begin{itemize} \item setting the default font size to 12pt and specifying `article' type for formatting; \item using the Palatino typeface and some special formatting for URIs (web addresses); \item preventing sections being numbered; \item turning off justification for an informal document; \item formatting a section heading; \item using the \LaTeX\ logo; \item generating today's date; \item formatting this list of items; \item formatting a subsection heading; \item using opening and closing quotes; \item formatting a URI; \item arbitrary formatting: centering and italicisation; \item autonumbering the pages. \end{itemize} \subsection{More information} This example was taken from the book `Formatting Information', which you can read online or in PDF format at \url{http://latex.silmarii.ie/formattinginformation/} and use as a teach-yourself guide. \begin{center} \fbox{\textit{Have a nice day!}} \end{center} \end{document}\end{verbatim}
1. **Install \LaTeX** on your computer

   See Appendix 1 starting on page 189. You can check to see if it’s already installed by opening a command window and typing `\$latex` and pressing the Enter or Return key; or by looking for a \LaTeX editor in your Programs or Applications, or in your dock or panel.

2. Make sure your editor is set up to use \TeX\, not `pdflatex` or the original \LaTeX. This setting is usually in the configuration; in `\LaTeX Studio` it’s under `Options > Configure \LaTeX Studio > Build Default Compiler` (see Figure 1.3 on the next page);

3. **Create a new, empty document**

   Start up your \LaTeX editor and open a new document:

   Click on `File > New` and if it offers you a choice of document types, pick ‘Empty File’.

   Delete any template material it inserts, so that your new document is completely empty.

4. **Copy the example**

   Copy and paste the text from Figure 1.2 on the preceding page. Make sure you get all of it (apart from the caption), and don’t change anything yet.

   If you are using the PDF or print edition of this book, copy the text from the web site.

5. **Save the document**

   Save the document as `quickstart.tex` in your `Documents` folder (or wherever you normally keep your documents);

6. **Typeset the document**

   Click on the `Run`, `Build`, `Typeset`, or `\TeX > File` menu; or on the toolbar icon for your editor, as indicated by the black arrow in the illustrations in Figure 1.4 on page 10.

7. **Preview the typesetting**
1.4 \LaTeX{} commands

Now that you have seen \LaTeX{} working, let’s have a closer look at what it’s actually doing. \LaTeX{} commands all begin with a backslash (\) and normally consist of lowercase letters only (there are a few which have uppercase letters). Going through the quickstart.tex
Figure 1.4: What to click on to typeset a document

The *Kile* editor (Linux, Mac, Windows)

The *TEXshop* editor (Mac)

The *TEXStudio* editor (Linux, Windows, Mac)

The *Emacs* editor (Linux, Windows, Mac)

The *\LaTeX* Editor app (Android)

If you are using another editor, look for a menu or toolbar button marked \Typeset\ or \XeLaTeX\ or \PDFLaTeX\ or \Build\ or \Compile\.
document in Figure 1.2 on page 7, we can see the following commands being used:

\documentclass specifies the class of document (article) and the size of type for the text (12pt);

\usepackage tells \LaTeX to use the named packages (plugins), here palatino (a font) and url (provides a way to format URIs);

\setcounter sets the value of a counter, here secnumdepth, to a value, here zero, which in this case prevents sections being numbered);

\begin marks the beginning of an environment, here document, which contains the whole text of the document. It’s terminated by a matching \end{document} command further down.

Everything up to this point has been a preamble, setting up how the document looks;

\section identifies a section heading;

\LaTeX typesets the \LaTeX logo;

\today typesets today’s date;

\begin marks the beginning of an itemize environment, which is an itemized (bulleted) list. It’s terminated by a matching \end{itemize} command at the end of the list;

\item marks the start of a new list item;

\end ends an environment, here the itemized list;

\subsection identifies a subsection heading;

\url typesets a URI, allowing line-breaks only at slashes or dots;

\begin begins another environment, center, which centres the material;

\fbox typesets the material in curly braces in a framed box;

\textit typesets the material in curly braces in italic type;

\textit typesets the material in curly braces in italic type;
\end{center} environment;
\end{document} ends the document environment, and thereby terminates the document.

Backslashes and forward slashes
Do not confuse the backslash (\) with the forward slash (/). They are two different characters.

- The forward slash is used in Unix-based systems (including Mac OS X and GNU/Linux) to separate directory names and file names;
- The forward slash is also used on the Web to separate the directory names and file names in a URI;
- The backslash is used to separate directory names and file names in the Microsoft Windows file system only.

The backslash is used to signal the start of a \LaTeX command in all systems, and when you refer to directory and file names in \LaTeX (e.g., image files), you MUST use the forward slash, even in Microsoft Windows.

1.4.1 Simple commands
Simple commands are just the command name on its own, after the backslash, for example:

\today

This example is an instruction to \LaTeX to insert the current date at that point. You would usually use this in a draft article or report somewhere close to the beginning, so that you have a record of when it was last typeset. You don't have to do anything else, although there are packages for changing the format of the date.

You will also come across several font-changing commands like \sffamily (switches to the sans-serif font family); \bfseries (switches to the boldface font series); \Large (steps up to the Large
1.5. WHITE-SPACE IN \LaTeX

size of type); but you should wait until you have gone through Chapter 6
starting on page 129 before trying to use them, as there are other forms
of the commands more suited to textual use.

1.4.2 Commands with arguments

Most \LaTeX commands are followed by one or more arguments, meaning
information to be acted upon. Here are two examples, a chapter
title (see section 2.6 on page 47) and a cross-reference label (see section 5.3.1 on page 110):

\chapter{Poetic Form} \label{pform} The shape of poetry
when written or printed distinguishes it from prose.

Such arguments always go in \{ curly braces \} like those shown above.
Be careful not to confuse the curly braces on your keyboard with
(round) parentheses, [square] brackets, \textless less-than or greater-than \textgreater
signs, \langle angled \rangle brackets, or «guillemets» (French quotes, not guillemots;
those are sea-birds). They are all quite different and they mean
different things, as shown.

1.5 White-space in \LaTeX

In \LaTeX documents, all multiple consecutive spaces and TAB characters
are treated as if they were a single space during typesetting. All multiple
newlines (linebreaks) are treated as if they were just two newlines (a
paragraph break).

\LaTeX does its own spacing and alignment using the commands you
give it and the layout in the stylesheet, so you have extremely precise
control. You are therefore free to use extra white-space in your editor for
optical ease and convenience when editing.

1.5.1 Swallowing space after commands

Rule: \LaTeX discards any white-space after a command ending in a letter
when there is no argument present.

It does this automatically, so that you don’t get unwanted extra
space in your typeset output, but it does mean that any simple

2 Embarrassingly, the \LaTeX command for guillemets was mis-spelled guillemot when it
was created, and no-one seems to have the nerve to change it. Albatross!

Formatting Information
command which ends in a letter and has no argument must be followed by white-space or an empty pair of curly braces before the text which follows it, to keep it separate. Read that again.

\tableofcontents Thanks to Aunt Mabel for all her help with this book.

\tableofcontents Thanks to Aunt Mabel for all her help with this book.

\tableofcontents Thanks to Aunt Mabel for her help with this book.

\tableofcontents{} Thanks to Aunt Mabel for all her help with this book.

The additional spacing or braces is not needed if the command name ends with a non-letter, or is directly followed by another command, or occurs immediately before a closing curly-brace, or is followed by a double newline (paragraph break). Read that again, too.

Simple one-word commands (like \tableofcontents) must therefore be separated from any following text with curly braces or white-space, which means a normal space or a newline (linebreak) or a TAB character. If you forget the white-space, like this:

\tableofcontentsThanks to Aunt Mabel for all her help with this book.

then \LaTeX will treat everything up to the next space as a command, and end up trying to make sense of a ‘command’ apparently called \tableofcontentsThanks. There’s no such command, of course, so \LaTeX will complain by displaying an error message (see section C.3.3.2 on page 252).

With commands that take arguments you do not need to use extra white-space or curly braces after the command, because the existing curly braces will keep the command separate from any normal text which comes after it. The following example is therefore exactly equivalent to the one we just saw in section 1.4.2 on the preceding page, and will typeset identically despite the absence of spaces between commands.
1.6 SPECIAL CHARACTERS

The shape of poetry when written or printed distinguishes it from prose.

By the same token, the following example is therefore also exactly equivalent (although rather unusual!):

The shape of poetry when written or printed distinguishes it from prose.

That is, it will get typeset exactly the same.

Why would you want such odd spacing (or none)? The answer is usually never, although extra blank lines in your editor between chapters or sections make editing easier. But a lot of \LaTeX is not typed by hand: it is generated by computer programs from other systems such as web scripts, XML documents, databases, filestores, mashup engines and other processes, and it makes life easier for the programmers if they don’t have to worry about the odd space or two creeping in here and there in normal text: it simply won’t have any effect. It also means that if you want to use extra spacing to make your text easier to edit, you don’t have to worry about unwanted linebreaks coming out between sections or paragraphs, tabbing in tables, or indentation in list items.

1.6 Special characters

There are ten keyboard characters which have special meanings to \LaTeX, and cannot be used on their own except for the purposes shown in Table 1.1 on the following page.

These characters were deliberately chosen, either because they are rare in normal text, or (in the case of $, #, &, and %) they already had an established special meaning on computers as metacharacters (characters standing as symbols for something else) when \TeX was written.

1.6.1 Using the special characters

We saw at the start of this section how to use the backslash to begin a command, and how to use curly braces to delimit an argument, and we
### Table 1.1: Special characters in LATEX

<table>
<thead>
<tr>
<th>Key</th>
<th>Special meaning</th>
<th>If you need the actual character itself, type it like this:</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\</code></td>
<td>The command character</td>
<td>textbackslash()</td>
<td></td>
</tr>
<tr>
<td><code>$</code></td>
<td>Math typesetting delimiter</td>
<td>$$</td>
<td>$$37.46</td>
</tr>
<tr>
<td><code>%</code></td>
<td>The comment character</td>
<td>%</td>
<td>42 %</td>
</tr>
<tr>
<td><code>^</code></td>
<td>Math superscript character</td>
<td>(^{})</td>
<td>(^{})</td>
</tr>
<tr>
<td><code>&amp;</code></td>
<td>Tabular column separator</td>
<td>&amp;</td>
<td>AT&amp;T</td>
</tr>
<tr>
<td><code>_</code></td>
<td>Math subscript character</td>
<td>_(_)</td>
<td>total_A</td>
</tr>
<tr>
<td><code>-</code></td>
<td>Non-breaking space</td>
<td>~</td>
<td>~{}</td>
</tr>
<tr>
<td><code>#</code></td>
<td>Macro parameter symbol</td>
<td>#</td>
<td>#42</td>
</tr>
<tr>
<td><code>{</code></td>
<td>Argument start delimiter</td>
<td>{</td>
<td>${$</td>
</tr>
<tr>
<td><code>}</code></td>
<td>Argument end delimiter</td>
<td>}</td>
<td>$} $</td>
</tr>
</tbody>
</table>

are not covering the mathematical uses in this book. The only special characters remaining from the list in Table 1.1 are therefore:

- **The comment character** makes LATEX ignore the remainder of the line in your document, so you can see it in your editor, but it will never get typeset. For example:

  ```latex
  Today's price per kilo is €22.70
  % get Mike to update this daily
  ```

  As with all comments in documents, don’t forget to remove them before sending the original document source to someone else!

- **The tilde** in LATEX prints as a normal space, but prevents a linebreak ever occurring at that point. It’s often used between a person’s initials and their surname, such as P.~Flynn, in case it might be

```latex
Formatting Information
```
1.7 QUOTATION MARKS

Typeset towards the end of a line where an intervening linebreak would make it harder to read.

The **ampersand** is used in tabular setting (rows and columns) to separate the cell values within each row. We’ll see how this works in section 4.2 on page 79.

The **hash mark** or octothorpe is the American ‘pound’ [weight] or ‘number’ sign. For a pound (sterling) sign (£, now nearly obsolete except in the UK and some of its former dependencies, and for historical purposes), use the `\textsterling` command.

While we’re on the subject of money, not every font has a Euro character (€), especially those designed before the currency was invented. The default € sign in many fonts is a crummy design based on the letter C instead of the rounded E. An official (sans-serif) Euro sign € is in the `marvosym` package and is done with the `\EUR` command (see section 3.1 on page 56 for details of how to use \LaTeX packages). However, a slightly unusual but more interesting serif Euro sign € is in the `textcomp` package using the `\texteuro` command with the the TS1 font encoding (see section 6.2.5 on page 147 and section 6.2.4 on page 146 for details of switching typefaces and settings in mid-text).

1.7 Quotation marks

If you are using Xe\LaTeX and UTF-8, you can use your operating system’s curly ‘open-quote’ and ‘close-quote’ characters.

Otherwise, use the [“] key (grave-accent or ‘backtick’) for the opening quote, and the [’] (apostrophe) key for the closing quote, doubled if you want double quotes; \LaTeX will automatically typeset these as real quotes:

```
He said, "I'm just going out."
```
CHAPTER 1. WRITING DOCUMENTS

Ordinal superscripts

Don’t use them in normal English text. Superscripted ordinals like 21st are a historical relic of Victorian and earlier typography. Unless done with skill and finesse by a typographer, they are usually ugly and unnecessary, and are never used in modern professional typesetting. They were re-introduced by Microsoft Word apparently because some American corporations liked their wordprocessing to look like what they fondly imagine it used to be.

If you want to try and mimic low-quality wordprocessing, or if you are trying to make a genuine typographic facsimile of antiquarian typesetting, use the \textsuperscript command from the textcomp package. Never use math mode superscripting for text superscripts — it’s the wrong height, wrong size, and possibly the wrong font.

In non-English languages (European, at least) it is completely different: the ordinal feminine and masculine (like 2ª ‘secunda’ or 8º ‘octavo’ is the normal method of representation, and their use in Latin-derived terminology in (eg) printing and binding remains standard.

Do not use the unidirectional typewriter single-quote ‘ key (apostrophe) or double-quote ” key (quotes) for opening quotes: \LaTeX treats these as closing quotes only.

However, if you are using Emacs as your editor, the ‘ key is specially programmed in latex-mode to think for itself and produce correct ‘ ‘ and ” characters automatically.

When typing one quotation inside another, there is a special command \thinspace which provides just enough separation between double and single quotes (leaving no space would make them look like triple-quotes, and using a normal space is too much and could allow an unwanted linebreak):

```
He said, “Her answer was “never” \thinspace', and she meant it.
```

```
He said, ‘Her answer was “never”’, and she meant it.
```
1.8 Accents

For accented letters in Latin-alphabet languages, use the accented keys from your keyboard. If you don’t have any, use your computer’s or editor’s character map (usually under the Insert Special Characters menu) to pick them from the pop-up window.

- If you are using \LaTeX{} with bibliography/bib, no further configuration is needed, as \LaTeX{} handles UTF-8 character encoding as the default;

- If you are using PDF\LaTeX{} with Bib\TeX{} you SHOULD always start your \texttt{Preamble} with these two lines to handle accents and other UTF-8 characters:

\begin{verbatim}
usepackage[utf8]{inputenc}
usepackage[T1]{fontenc}
\end{verbatim}

- If you are using PDF\LaTeX{} with bibliography/bib, you SHOULD do the same, but use the \texttt{utf8} option (no \texttt{x}) instead of \texttt{utf8x}:

\begin{verbatim}
usepackage[utf8]{inputenc}
usepackage[T1]{fontenc}
\end{verbatim}

That takes care of telling \LaTeX{} what character repertoire (\texttt{inputenc}, the input encoding) your system is using, and which set of fonts (\texttt{fontenc}, the ‘font encoding’) to find the extra characters in. \LaTeX{} needs no telling, except for one extra package for font-handling (see item section 1.8).
CHAPTER 1. WRITING DOCUMENTS

The Preamble

Modifications which you want to affect a whole document go at the start of your \LaTeX file, immediately after the \texttt{documentclass} line and before the \texttt{begin\{document\}} line. Here, for example, we want to rename the Abstract to 'Preview':

\begin{verbatim}
documentclass[a4paper]{report} 
renewcommand{\abstractname}{Preview} 
begin\{document\} ... 
end\{document\}
\end{verbatim}

This position, between the Document Class Declaration and the beginning of the \texttt{document} environment, is called the \textit{Preamble}, and it is used for modifications to the style and behaviour of this document.

Major or permanent modifications that you use in lots of documents all the time should probably go in a class file or package of your own making, but that’s beyond the scope of this book.

You also need to make sure your computer’s operating system is set to UTF-8, and that your editor is set the same way. Most are — with most modern systems, Unicode compatibility will let you use almost any letter or symbol from any writing system encoded in UTF-8 (the multibyte 8–bit encoding), for which \TeX has extensible support.\(^3\)

For language-specific hyphenation and cultural adaptation (including the correct language headings for all the parts of your document) use the \texttt{babel} package (see section 1.9.6 on page 30). For non-Latin typefaces you will also need the relevant font packages and typefaces (see section 6.2 on page 137).

Failing all this, if you don’t have accented letter keys on your keyboard, or you can’t find the codes to type, or if you need additional accents or symbols which are not in any of the keyboard tables, you can use the symbolic notation in Table 1.2 on page 22. In fact this can be used to put any accent over any letter (Welsh users can get a ŵ with

\(^3\) Note for M\TeX\ users: the \texttt{TeXShop} editor that comes with M\TeX is not set for UTF-8 by default: see step 23 on page 201 for how to set it.

Formatting Information
1.8. ACCENTS

If you don’t have accented letters on your keyboard

This is for users whose keyboards do not have native accent characters on them. See your Operating System manual for full details. Here are some common examples:

- On GNU/Linux systems the letter é is usually got with \texttt{AltGr+e}, depending on the keyboard setup that you installed. If you know the Unicode hexadecimal code-point (number) of the character you want, press \texttt{Ctrl+U} and release, followed by the number, and press Enter. Refer to the \texttt{xkeycaps} utility for a table of key codes and combinations (install it with your system’s package manager or get it from \url{www.jwz.org/xkeycaps/}).

- On Apple Mac OS X systems, the letter é is got with \texttt{Alt+e}.

- Under Microsoft Windows the letter é is got with \texttt{Ctrl+’+e}. If you know the Microsoft encoding (number) of the character you want, hold down the \texttt{Alt} key and type the number on the numeric keypad (not the top row of shifted numerals), then release the \texttt{Alt} key. Refer to the \texttt{charmap} utility for a table of key codes and combinations.

\texttt{\textasciitilde}, even for combinations which don’t actually exist in any language: if you particularly want a ģ, for example, you can have one with the command \texttt{\textasciitilde g}.

If you use this symbolic method only, you do not need to use the \texttt{inputenc} package described above.

Before the days of keyboards and screens with their own real accented characters, the symbolic notation was the only way to get accents, so you may come across a lot of older documents (and users!) using this method all the time: it does have the advantage in portability that the \LaTeX{} file remains plain \texttt{ASCII}, which will work on all machines
Table 1.2: Symbolic notation for accents

<table>
<thead>
<tr>
<th>Accent</th>
<th>Example</th>
<th>Characters to type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acute (fada)</td>
<td>é</td>
<td>'e</td>
</tr>
<tr>
<td>Grave</td>
<td>è</td>
<td>\e</td>
</tr>
<tr>
<td>Circumflex</td>
<td>é</td>
<td>^e</td>
</tr>
<tr>
<td>Umlaut or diaeresis</td>
<td>è</td>
<td>&quot;e</td>
</tr>
<tr>
<td>Tilde</td>
<td>ŋ</td>
<td>~n</td>
</tr>
<tr>
<td>Macron</td>
<td>ô</td>
<td>=o</td>
</tr>
<tr>
<td>Bar-under</td>
<td>o</td>
<td>\b o</td>
</tr>
<tr>
<td>Dot-over (sermû)</td>
<td>m</td>
<td>.m</td>
</tr>
<tr>
<td>Dot-under</td>
<td>ş</td>
<td>\d s</td>
</tr>
<tr>
<td>Breve</td>
<td>ũ</td>
<td>\u u</td>
</tr>
<tr>
<td>Háček (caron)</td>
<td>ŭ</td>
<td>\v u</td>
</tr>
<tr>
<td>Long umlaut</td>
<td>ō</td>
<td>\H o</td>
</tr>
<tr>
<td>Tie-after</td>
<td>Ţ Ţ</td>
<td>\t oo</td>
</tr>
<tr>
<td>Cedilla</td>
<td>č, Ć</td>
<td>\c c, \c C</td>
</tr>
<tr>
<td>O-E ligature</td>
<td>œ, Ő</td>
<td>\oe, \OE</td>
</tr>
<tr>
<td>A-E ligature</td>
<td>æ, Å</td>
<td>\ae, \AE</td>
</tr>
<tr>
<td>A-ring</td>
<td>á, Â</td>
<td>\aa, \AA</td>
</tr>
<tr>
<td>O-slash</td>
<td>o, Ø</td>
<td>\o, \O</td>
</tr>
<tr>
<td>Soft-l</td>
<td>ł, Ł</td>
<td>\l, \L</td>
</tr>
<tr>
<td>Ess-zet (scharfes-S)</td>
<td>Ÿ</td>
<td>\ss</td>
</tr>
</tbody>
</table>

everywhere, regardless of their internal encoding, and even with very old \TeX\ installations.\footnote{Remember not everyone is lucky enough to be able to install new software: many users on business and academic networks still use old versions of \TeX\ because they or their system managers don’t know how to update them. Local user groups may be able to provide help and support here.}

Irish and Turkish dotless-ı can be done with the special command \i, so an ı (which is normally typed with ı) may require \i{ı} if you need to type it in the long format — remembering that dummy pair of curly braces if there is no punctuation, because of the rule that \TeX\ control sequences which end in a letter (see section 1.5.1 on page 13) always absorb any following space. So what you might see as Rı Team ˙ rač would have to be R\i\ Team\.mra\.c when typed in full (there are not usually any dedicated keyboard keys for the dotless-ı}
or for aspirated or lenited characters). A similar rule applies to dotless-\j and to uppercase \i.

Note that modern versions of \LaTeX\ can compensate for this when used with the \texttt{utf8x} option of the \texttt{inputenc} package and the \texttt{T1} option of the \texttt{fontenc} package (as shown above). In that case you can just type \texttt{f\textquoteface{is}} to get ‘fís’. If you use \texttt{XeLaTeX} you don’t even need those packages and options, and you just type ‘fís’.

1.9 Dimensions, hyphenation, justification, and breaking

\LaTeX\’s internal measurement system is extremely accurate. The underlying \TeX\ engine conducts all its business in units smaller than the wavelength of visible light, so if you ask for 15mm space, that’s what you’ll get — within the limitations of your screen or printer, of course. While modern high-resolution displays use pixels smaller than you can easily see, many older screens cannot show dimensions of less than $\frac{1}{96}$” without resorting to magnification or scaling; and on printers, even at 600dpi, fine oblique lines or curves can still sometimes be seen to stagger the dots.

At the same time, many dimensions in \LaTeX\’s preprogrammed formatting are specially set up to be flexible: so much space, plus or minus certain limits to allow the system to make its own adjustments to accommodate variations like overlong lines, unevenly-sized images, and non-uniform spacing around headings.

\TeX\ uses a very sophisticated justification algorithm to achieve a smooth, even texture to normal paragraph text by justifying a whole paragraph at a time, quite unlike the line-by-line approach used in most wordprocessors and DTP systems.

Occasionally, however, you will need to hand-correct an unusual word-break or line-break, and there are facilities for doing this on individual occasions as well as automating it for use throughout a document.

1.9.1 Specifying size units

Most people in the printing and publishing businesses in English-speaking cultures habitually use the traditional printers’ points, picas
and ems as well as cm and mm when dealing with clients. Many older English-language speakers (and most North Americans) still use inches. In continental European and related cultures, Didot points and Ciceros (Didot picas) are also used professionally, but cm and mm are standard everywhere else: inches are only used now when communicating with North American cultures.

You can specify lengths in \LaTeX{} in any of these units, plus some others (see Table 1.3 on the next page).

The em can cause beginners some puzzlement because it’s based on the ‘point size’ of the type, which is itself misleading. The point size refers to the depth of the metal body on which foundry type was cast in the days of metal typesetting, not the printed height of the letters themselves (see Figure 1.5). Thus the letter-size of 10pt type in one typeface can be radically different from 10pt type in another (look at Figure 1.6 on the next page, where the widths are given for 10pt type). An em is the height of the type-body in a specific size, so 1em of 10pt type is 10pt and 1em of 24pt type is 24pt. A special name is given to the 12pt em, a ‘pica’ em, and a pica has become a

Figure 1.5: Some parts of a piece of metal type

Based on an original image by Victor Puebla (mimoriarty.wordpress.com/about/)
1.9. DIMENSIONS, HYPHENATION, JUSTIFICATION, AND BREAKING

Figure 1.6: An M of type of different faces boxed at 1em

The red line is the common baseline. Surrounding letters in grey
are for illustration of the actual extent of the height and depth of
one em of the current type size.

fixed measure in its own right. An old name for a 1em space is a
‘quad’, and \TeX has a command \texttt{\textbackslash quad} for leaving exactly that much
horizontal space.

Table 1.3: Units in \TeX

<table>
<thead>
<tr>
<th>Unit</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>pt</td>
<td>Anglo-American standard points (72.27 to the inch)</td>
</tr>
<tr>
<td>pc</td>
<td>pica ems (12pt)</td>
</tr>
<tr>
<td>bp</td>
<td>Adobe’s ‘big’ points (72 to the inch)</td>
</tr>
<tr>
<td>sp</td>
<td>\TeX’s ‘scaled’ points (65,536 to the pt)</td>
</tr>
<tr>
<td>dd</td>
<td>Didot (European standard) points (67.54 to the inch)</td>
</tr>
<tr>
<td>cc</td>
<td>Ciceros (European pica ems, 12dd)</td>
</tr>
</tbody>
</table>
| em        | ems of the current point size (historically the width of a letter
            ‘M’ but see below)                                              |
| ex        | x-height of the current font (height of letter ‘x’)                 |
| cm        | centimeters (2.54 to the inch)                                       |
| mm        | millimeters (25.4 to the inch)                                       |
| in        | inches                                                              |

To highlight the differences between typefaces at the same size, Figure 1.6 shows five capital Ms in different faces, surrounded by a box exactly 1em of those sizes wide, and showing the actual width of each M when set in 10pt type. Because of the different ways in which typefaces are designed, none of them is exactly 10pt wide.
If you are working with other DTP users, watch out for those who think that Adobe points (bp) are the only ones. The difference between an Adobe big-point and the standard point is only .27pt per inch, but in 10" of text (a full page of A4) that's 2.7pt, which is nearly 1mm, enough to be clearly visible if you're trying to align one sample with another.

1.9.2 Hyphenation

\LaTeX\ hyphenates automatically according to the language you use (see section 1.9.6 on page 30). To specify different breakpoints for an individual word, you can insert soft-hyphens (discretionary hyphens), done with the \- command (backslash-hyphen) wherever you need them, for example:

\begin{verbatim}
When in Mexico, we visited Popo-ca-tépetl by helicopter.
\end{verbatim}

If the words needs to be hyphenated, the best-fit of the points will be used, and the rest ignored.

To specify hyphenation points for all occurrences of a word in the document, use the \hyphenation command in your Preamble (see the panel ‘The Preamble’ on p. 20) with one or more words as patterns in its argument, separated by spaces. This will even let you break ‘helicopter’ correctly. In this command you use normal hyphens in the pattern, not soft-hyphens.

\begin{verbatim}
\hyphenation{helico-pter Popo-ca-tépetl vol-ca-no}
\end{verbatim}

If you have frequent hyphenation problems with long, unusual, or technical words, ask an expert about changing the value of \spaceskip, which controls the flexibility of the space between words. This is not something you would normally want to do without advice, as it can change the appearance of your document quite significantly.

If you are using a lot of unbreakable text (see the next section and also section 4.7.1 on page 102) it may also cause justification problems. One possible solution to this is shown in section 7.3 on page 165.
1.9.3 Breakable and unbreakable text

Unbreakable text is the opposite of discretionary hyphenation. To force \LaTeX to treat a word as unbreakable, use the \mbox command:

\mbox{pneumonoultramicroscopicsilicovolcanoconiosis}

This may have undesirable results, however, if you subsequently change margins or the size of the text: pneumonoultramicroscopicsilicovolcanoconiosis, although if you’re reading this in a browser, you probably won’t see the effect properly: look at the PDF.

Another option, for reoccurring words, is to use the \hyphenation command as shown in section 1.9.2 on the facing page, but give the word[s] with no hyphens at all, which stops them having any break-points.

To tie two words together with an unbreakable space (hard space), use a tilde (~) instead of the space (see the list in section 1.6.1 on page 15). This will print as a normal space but \LaTeX will never break the line at that point.

A normal space between words is always a candidate for a place to break the text into lines, and the word-spacing gets evened-out between all the remaining words...with one exception: a full point (period) after a lowercase letter is treated in \LaTeX as the end of a sentence, and it automatically puts a little more space before the next word. You do not (and should not) type any extra space yourself.

However, after abbreviations in mid-sentence like ‘Prof.’, it’s not the end of a sentence, so we need a way to tell \LaTeX that this should be a normal space. The command for doing this is \␣ (backslash-space — I have made the space visible here so you can see it, but it’s just a normal space). This prevents \LaTeX from adding the extra sentence-space and it also means it becomes a normal breakpoint (otherwise you would use the tilde as described above).

For example, it would look wrong to break the name Prof. D.E. Knuth at a line-end. It’s a good idea to make this standard typing practice for things like people’s initials followed by their surname, as Prof.\␣D.E.~Knuth.
1.9.4 Dashes

The hyphen (−) is only used for hyphenated compound words like editor-in-chief. \LaTeX{} inserts its own hyphens when it needs to break a word at right right-hand margin.

Dashes are different: they’re longer and they are used in different places. Check the panel ‘If you don’t have accented letters on your keyboard’ on p. 21 for how to find these characters in your computer’s character-map.

**Long dash**: The long dash — what printers call an ‘em rule’ like this — is used to separate a short phrase from the surrounding text in a similar way to parentheses. If you’re using \LaTeX{}, you can just type the long dash on your keyboard.

- If you can’t find the character, type three hyphens typed together, like --- this: \LaTeX{} will recognise this combination and replace it with a real em rule.
- If you want space either side, bind the first hyphen to the preceding word with a tilde like ~--- this and use a normal space after the third hyphen (shown as a visible space here, but it’s just a normal space). This avoids the line being broken before the dash.

The difference between spaced and unspaced rules is purely æsthetic. Never use a single hyphen for this purpose.

**Short dash**: The short dash is used between digits like page ranges (35–47). Printers call this an ‘en-rule’ and if you’re not using \LaTeX{} you can get it by typing two hyphens together, as in 35--47. Never use a single hyphen for this purpose either.

**Minus sign**: If you want a minus sign, use math mode (see section 1.10 on page 31) where you type a normal hyphen between math delimiters like \((x=y-z)\). Don’t use the hyphen for a minus sign outside math mode.

There are other dashes for special purposes in the Unicode repertoire, but they are out of scope for this document.
1.9.5 Justification

The default mode for typesetting in \LaTeX{} is justified (two parallel margins, with word-spacing adjusted automatically for the best optical fit). In justifying, \LaTeX{} will never add space between letters, only between words. The soul package can be used if you need letter-spacing (‘tracking’), but this is best left to the expert.

There are two commands \texttt{\raggedright} and \texttt{\raggedleft} which typeset with only one margin aligned. Ragged-right has the text ranged (aligned) on the left, and ragged-left has it aligned on the right. They can be used inside a \texttt{group} (curly-braces, for example: see the panel ‘Grouping’ on p. 148) to confine their action to a part of your text, or put in the Preamble if you want the whole document done that way. This paragraph is set ragged-right.

These modes also exist as environments called \texttt{raggedright} and \texttt{raggedleft} which are more convenient when applying this formatting to a whole paragraph or more, like this one, set ragged-left.

\begin{raggedleft}
These modes also exist as environments called \texttt{raggedright} and \texttt{raggedleft} which is more convenient when applying this formatting to a whole \texttt{paragraph} or more, like this one.
\end{raggedleft}

Ragged setting turns off hyphenation. There is a package \texttt{ragged2e} providing the command \texttt{\RaggedRight} (note the capitalisation) which retains hyphenation in ragged setting, useful when you have a lot of long words. There’s a \texttt{\RaggedLeft}, too.

To centre text, which is in effect both ragged-right and ragged-left at the same time, use the \texttt{\centering} command inside a \texttt{group}, or use the \texttt{center} environment.

Be careful when centering headings or other display-size material, and add manual linebreaks where needed (\textbackslash\textbackslash) to make the breaks at sensible pauses in the meaning (Flynn, 2012). \textit{Never} rely on the automated line-breaking of editors in these cases.
1.9.6 Languages

\LaTeX can typeset in the native manner for several dozen languages. This affects hyphenation, word-spacing, indentation, and the automatic names of the parts of documents displayed in headings (but not the commands used to produce them).

Most distributions of \LaTeX come with US English and one or more other languages installed by default, but it is easy to use the babel package and specify any of the supported languages or variants, for example:

\begin{verbatim}
\usepackage[german,frenchb,english]{babel}
...
As one writer has noted, \selectlanguage{german}``Das berühmte Voltaire-Zitat, \emph{\foreignlanguage{frenchb}{il est bon de tuer de temps en temps un amiral pour encourager les autres}}), ist ein Beispiel sarkastischer Ironie.''
\end{verbatim}

Make sure that the base language of the document comes last in the list. The list of supported languages is in the package documentation.

Changing the language with babel is a cultural shift: it changes the hyphenation patterns, word-spacing, the way in which indentation is used, and the names of the structural units and identifiers like ‘Abstract’, ‘Chapter’, and ‘Index’, etc. For example, using French as the default, chapters will start with ‘Chapitre’.

The selectlanguage lets you tell \LaTeX when to switch to the language specified in the argument. If you have only a small fragment in another language (a word or two, maybe a sentence, but less than a paragraph), you can use the command foreignlanguage to change the language just for that text. The first argument gives the language; the second contains the word or phrase.

The babel package uses the hyphenation patterns provided with your version of \LaTeX (see the start of your document log files for a list). For other languages you need to set the hyphenation separately (outside the scope of this book).
1.10 Mathematics

As explained in p. xvi, \TeX{} was originally written to automate the typesetting of books containing mathematics, and mathematics is typeset differently from normal text. This book does not cover mathematical typesetting, which is explained in detail in many other books and Web pages, so all we will cover here is the existence of the math mode commands, and some characters which have special meaning, so they don’t trip you up elsewhere.

In addition to the 10 special characters listed in section 1.6 on page 15, there are three more characters which only have any meaning inside mathematics mode:

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>|</td>
<td>Vertical bar</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less-than</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater-than</td>
</tr>
</tbody>
</table>

If you type any of these in normal text (that is, outside math mode), you will get very weird things happening and lots of error messages. If you need to print these characters, you must type them using math mode, or use the symbolic names from the textcomp package (\textbrokenbar, \textl angle, and \textrangle).

The hyphen also has a different meaning in math mode, as we saw in the previous section: it typesets as a minus sign, so if you want to write about negative numbers outside math mode, you need to type the number using inline math delimiters.

\[
\bar{n}^{*}_{j}(s) = \frac{\sum_{i=1}^{k} \left\{ n_{i}(0)p_{i,k+1}^{*}(s) + M^{*}(s) \right\}}{\sum_{i=1}^{k} p_{i}p^{*}_{i,k}^{*}(s)} + \sum_{i=1}^{k} n_{i}(0)p_{i,j}^{*}(s), \quad (j = 1, 2, \ldots, k).
\]

To use math mode inline (that is, within a paragraph), enclose your math expression in \( \) (and \) commands (round parentheses). You can get the much-quoted equation \( E = mc^2 \) by typing \( \{ E=mc^2 \} \), and to get a temperature like \( -40^\circ \) you need to type \( \{ -40^\circ \} \) in
order to get the minus sign and the right spacing. Never use the math superscript and a letter o for degrees: all that gets you is a raised italic letter o.

To typeset a math expression as ‘displayed math’ (centered between paragraphs, like the huge equation above), enclose it in the commands \[ and \] (square brackets).

Displayed equations can be auto-numbered with the equation environment instead of the \[ and \] commands.

---

5 Bear in mind that the degree symbol is a non-ASCII character, so you must specify what input encoding you are using if you want to type it: see the example of the inputenc package in section 1.8 on page 19. If you don’t want to use non-ASCII characters (or if you are using a system which cannot generate them), you can use the command \textdegree to get the degree sign.

6 You will also see dollar signs used for math mode. This is quite common but deprecated: it’s what plain \TeX used in the days before \LaTeX, and the habit got ingrained in many mathematicians. It still works as a convenient shorthand like \$x=\gamma\$, as do double-dollars for display-mode math like \$$E=mc^2\$$, but they are only mentioned here to warn readers seeing them in other authors’ work that \(...\) and \[ ...\] are the proper \LaTeX commands.
Basic structures

If the quick-start exercise in section 1.3 on page 7 was enough to show you how a \LaTeX{} document works, then this is where you get the rest of the basic information. If you skipped Chapter 1 starting on page 1 then be prepared to go back to some of the sections in it, because I'll be referring to things you might not have come across yet.

**What's what**

All you need to do in \LaTeX{} is to say what's what by labelling things as what they are. If you want a list, you say so. It's *not* a bunch of paragraphs prefixed with bullets, as in a wordprocessor, it's a list.

One of the small mind-shifts needed to work with a markup system like \LaTeX{} is that you need to be *explicit*. Unlike a wordprocessor, where the WYSIWYG display is the only way of communicating your intentions, with \LaTeX{} you can actually *tell* it what to do.

\LaTeX{}'s approach to formatting is to aim for *consistency*. This means that as long as you identify each component element of your document correctly, it will be typeset in the same way as all the other elements like it, so that you achieve a consistent finish with minimum effort.

Consistency helps make documents easier to read and understand, as well as making them more visually attractive. Consistency is also
what editors, reviewers, and publishers look for. Publishers have a house style, and often a reputation to keep, so they rightly insist that if you do something a certain way once, you should do it the same way each time.

‘Elements’ are the component parts of a document: all the pieces which make up the whole. Almost everyone who reads books, newspapers, magazines, reports, articles, and other classes of documents will be familiar with the common elements: parts, chapters, sections, subsections, subsubsections, headings, titles, subtitles, paragraphs, lists, tables, figures, and so on, even if they don’t consciously think about them.

2.1 The Document Class Declaration

In order to set things up correctly, \LaTeX{} needs to know up front what type of document you are going to be writing. There are probably lots of different types of document you deal with: in \LaTeX{} they are called ‘classes’ of documents — ‘class’ is just the computing science word for ‘type’.

2.1.1 Document classes

To tell \LaTeX{} what class of document you are going to create, the first line of your file MUST identify it.\footnote{Readers familiar with SGML, HTML, and XML will recognise the concept as similar to the Document Type Declaration (it’s still called a ‘type’ there, not a ‘class’).} To start a report, for example, you would type the \texttt{\documentclass} command like this as the first line of your document:

\begin{verbatim}
\documentclass{report}
\end{verbatim}

There are four built-in classes provided, and many others that you can download (some may already be installed for you):

\texttt{report} for business, technical, legal, academic, or scientific reports;

\texttt{theses}\footnote{Theses and dissertations may require an Abstract, which is provided in the report class but not in the book class. Many universities provide a special \texttt{thesis} class of their own.}, dissertations;
2.1. THE DOCUMENT CLASS DECLARATION

**article** for white papers, magazine or journal articles, reviews, conference papers, essays, or research notes;

**book** for books, booklets, or whole journals;

**letter** for letters.³

These default classes are fairly basic in terms of layout and design, in order to make them easier to customise by adding packages, which are the style and layout plug-ins that \LaTeX uses to let you automate formatting.

The **article** class in particular can be used (some would say ‘abused’) for almost any short piece of typesetting by simply omitting the titling and layout (see section 2.3 on page 42) and adding the relevant packages — like we saw in section 1.3 on page 7.

### 2.1.2 Extending the default classes

The built-in classes are intended as starting-points, especially for drafts, and for compatibility when exchanging documents with other \LaTeX users, as they come built into every installation of \LaTeX and are therefore guaranteed to format identically everywhere. They are *not* intended as final-format publication-quality layouts unmodified, and should never be used as such. For most other purposes, especially for publication, you use \LaTeX packages to extend these classes to do what you need. The most common ways to do this are:

- The **memoir** package and the **komascript** bundle contain more sophisticated replacements for all the built-in classes, as well as additional ones;

- Many academic and scientific publishers provide their own special class files for articles and books (on their Web sites for download);

- Conference organisers may also provide class files for authors to write papers for presentations;

- Many universities provide their own thesis document class files in order to ensure exact fulfilment of their formatting requirements (many of these are on CTAN);

³ The built-in **letter** class is rather idiosyncratic: there are much better ones you can use which you will find in the **memoir** package and the **komascript** bundle.
Businesses and other organisations can provide their users with corporate classes on a central server and configure \TeX installations to look there first for packages, fonts, etc (not usually available to the public, of course);

There are over 195 document classes on CTAN (see www.ctan.org/topic/class).

Books and journals are not usually printed on office-size paper. Although for draft purposes \TeX's layouts fit on the standard A4 or Letter stationery in your printer, it makes them look odd: the margins are too wide and the font size is too small, because the finished job will normally be trimmed to a completely different size entirely — try trimming the margins of the PDF version of this book to make it 188 mm × 235 mm (the same as the Companion series) and you'll be amazed at how it changes the appearance.

The four default built-in document classes are therefore adequate for drafts or for sending to a colleague to edit, but they are not really usable for final-format publishing. For this you need to add packages or to use a class file designed by your publisher or institution (or yourself!) to fit the type of publication. Quite often these are based on the default classes for compatibility, but typeset quite different output.

2.1.3 Document class options

The default layouts were originally designed to fit as drafts on US 'Letter' size paper.\footnote{Letter' size is 8½" × 11", which is the trimmed size of the long-obscure Demy Quarto, still in use in North America. Other common US office sizes are 'Legal', which is 8½" × 14", a 'bastard' (variant) cutting close to the old Foolscap (8¾" × 13¼"); Ledger or Tabloid (11" × 17", which is exactly twice 'Letter', in the same way that A3 is twice A4); and 'Executive' (7" × 10"). ISO standard 'A', 'B', and 'C' paper sizes, used everywhere else, are still largely unknown in most parts of North America. To create documents with similar margins for A4 paper, you need to specify the paper size in an optional argument in square brackets before the document class name, eg

\begin{verbatim}
\documentclass[a4paper]{report}
\end{verbatim}

Many \TeX systems now install the \texttt{a4paper} option as the default, so this may not be needed; on the contrary, North American users may
now need to specify the letterpaper option instead. The \texttt{geometry}
package, which we will see later, lets you specify other bigger and
smaller paper sizes.\footnote{Note that the standard built-in document classes
(book, article, report, or letter) only use the paper size to adjust the
margins: they do not embed the paper size name in the PostScript or PDF
output. For this you need the \texttt{geometry} package. If you are
using \texttt{PDF\LaTeX}, or you intend creating PostScript output, and you want
to change the default paper size, you \textbf{must} specify it both in the Document Class
option and as an option to the \texttt{geometry} package (see section 3.1.3 on
page 60), in order to ensure that the paper size name gets embedded correctly
in the output, otherwise printers may select the wrong paper tray, or reject the job.}

The other default settings are for:

1. 10pt type (all document classes);
2. two-sided printing (books and reports) or one-sided (articles and
   letters);
3. separate title page (books and reports only).

These can be modified with the following document class options which
you can add in the same set of square brackets, separated by commas
(the \texttt{10pt} option is the default):

\begin{itemize}
\item \texttt{11pt} to specify 11pt type (headings, footnotes, etc get scaled up or
down in proportion);
\item \texttt{12pt} to specify 12pt type (again, headings etc get scaled to match);
\item \texttt{oneside} to format one-sided printing for books and reports;
\item \texttt{twoside} to format articles for two-sided printing;
\item \texttt{titlepage} to force articles to have a separate title page;
\item \texttt{draft} makes \LaTeX{} indicate hyphenation and justification
problems with a small square in the right-hand margin of the problem line
so they can be located quickly by a human. This option also sets
graphics to print as an empty outline (rectangle) containing just the
filename of the image, so that image-heavy documents will print more
quickly and use less ink or toner.
\end{itemize}
CHAPTER 2. BASIC STRUCTURES

If you were using \LaTeX{} for a report to be in 12pt type on Letter paper, but printed one-sided in draft mode, you would use:

\begin{verbatim}
\documentclass[12pt,letterpaper,oneside,draft]{report}
\end{verbatim}

The 10pt, 11pt, and 12pt settings cover between them probably 99% of all common text-document typesetting. There are extra options for other body type sizes in the \texttt{extsizes} bundle of document classes (\texttt{extarticle}, \texttt{extbook}, \texttt{extreport}, etc), and various national and international organisations supporting the visually-impaired have special large-type document class options.

**Global options**

In addition to any options specific to the document class, it is also possible to put package options in the \texttt{\documentclass} options argument instead of in the \texttt{\usepackage} command (see section 3.1.2 on page 57), provided they are not implemented by more than one package. Packages which do not implement the named option at all are supposed to silently ignore it.

**Exercise 1. Create a new document**

1. Use your editor to create a new, empty document
   - If your editor insists on filling your new document with template material, delete it all so that the file is empty;
2. Type in a Document Class Declaration as shown above;
3. Add a font size option if you wish;
4. In North America, omit the \texttt{a4paper} option or change it to \texttt{letterpaper};
5. Save the file (make up a name) ensuring the name ends with \texttt{.tex}.
Picking suitable filenames

Never, never, NEVER create directories (folders) or file names which contain spaces or non-printing, non-ASCII characters. Although your operating system may support them, some don’t, and they will only cause grief and tears, especially in automation software like document builders, web scripts, and app-based remote compilers.

Make filenames as short or as long as you wish, but strictly avoid spaces. Stick to upper- and lower-case letters without accents (A–Z and a–z), the digits 0–9, the hyphen (–), the underscore (_), and the dot (full point or period: .) — similar to the conventions for a Web URI: it will let you refer to TEX files over the Web more easily, make your files more portable, and make it easier to use standard system utilities and applications, as well as those distributed with \TeX.

2.2 The document environment

After the Document Class Declaration, the text of your document is enclosed between two commands we saw in section 1.4 on page 9 which identify the beginning and end of the actual document (in the example below, you would put your text where the dots are):

\documentclass[11pt,a4paper,oneside]{report}
\begin{document}
...
\end{document}

The reason for marking the beginning of your document text is that \TeX allows you to insert extra setup specifications before it (where the blank line is in the example above: we’ll be using this soon). The reason for marking the end of your document text is to provide a place for \TeX to be programmed to do extra stuff automatically at the end of the document, like making an index.

A useful side-effect of marking the end of the document text is that you can store comments or temporary text underneath the \end{document} in the knowledge that \TeX will never see them and never try to typeset them (they don’t even need to be preceded by
the \% comment character), but they will remain in your document for you to see in your editor, or maybe to copy and paste in a later edit.

\end{document}

Don't forget to get the extra chapter from Jim!

This \begin{...}\end pair of commands is an example of a common LaTeX structure called an environment. Environments enclose text which is to be handled in a particular way. All environments start with \begin{...} and end with \end{...} (putting the name of the environment in the curly braces each time).

If you’re familiar with HTML, SGML, or XML you’ll recognise this technique: it’s just like start-tags and end-tags.
2.2. THE DOCUMENT ENVIRONMENT

Exercise 2. Add the document environment

1. Add the document environment to your new file;

2. If you are using PDFLaTeX (NOT if you are using XeLaTeX)...
   In between the Document Class Declaration and the \begin{document}, add the two lines we saw in section 1.8 on page 19 which allow the use of UTF-8 in PDFLaTeX:

   \usepackage[utf8]{inputenc}
   \usepackage[T1]{fontenc}

3. In the document environment, type the phrase Hello, World!

4. Save the file and typeset it; you should get some output like this:

   Hello, World!
CHAPTER 2. BASIC STRUCTURES

2.3 Titling

The first thing you actually put in the document environment is almost always the document title, the author's name, and the date (except in letters, which have a special set of commands for addressing). The title, author, and date are all examples of metadata (information about information).

\documentclass[11pt,a4paper,oneside]{report}
\begin{document}
\title{Practical Typesetting}
\author{Peter Flynn\ Silmaril Consultants}
\date{January 2016}
\maketitle
\end{document}

The \texttt{title}, \texttt{author}, and \texttt{date} commands are self-explanatory. You put the title, author name, and date in curly braces after the relevant command. The title and author are compulsory; if you omit the \texttt{date} command, \LaTeX{} uses today's date by default.

You MUST finish the metadata with the \texttt{maketitle} command, which tells \LaTeX{} that it's complete and it can typeset the titling information at this point. If you omit \texttt{maketitle}, the titling will never be typeset. This command is reprogrammable so you can alter the appearance of titles (like I did for the printed version of this document). It also means publishers can create new commands like \texttt{datesubmitted} in their own document classes, in the knowledge that anything like that done before the \texttt{maketitle} command will be honoured.

One extra command show here is the double backslash (\textbackslash\textbackslash), which is the \LaTeX{} command for a premature (forced) linebreak. \LaTeX{} normally decides by itself where to break lines, and it's usually right, but sometimes you need to cut a line short, like here, and start a new one. I could have left it out and just used a comma, so the name and company would all appear on the one line, but I just decided that I wanted the company name on a separate line. In some publishers’ document classes, they provide a special \texttt{affiliation} command to put your company or institution name in instead.
2.3. TITLING

Figure 2.1: Titling information typeset on the title page

The most common use of the double backslash in the \texttt{author} command is for separating multiple authors, so I don’t recommend that you do what I did here except for draft or experimental purposes.

When this file is typeset, you get something like Figure 2.1 (I’ve cheated and done it in \texttt{colour} for fun — yours will be in black and white for the moment). This is a report, so the title appears all by itself on a single page.

The order of the first three commands is not important, but the \texttt{maketitle} command must come last.

If you have mistyped a command, you may get an error message: see section C.3 on page 249 to resolve this.
Exercise 3. Adding the metadata

1. Add the \title, \author, \date, and \maketitle commands to your file.
2. Use your own name, make up a title, and give a date.
3. Typeset the document.

2.4 Abstracts and summaries

In reports and articles it is usual for the author to provide an Summary or Abstract, which describes the content and explains its importance. Abstracts in articles are usually only a few paragraphs long. Summaries in reports or theses can run to several pages, depending on the length and complexity of the document or the readership it’s aimed at.

\documentclass[11pt,a4paper,oneside]{report}
\begin{document}
\title{Practical Typesetting}
\author{Peter Flynn\ Silmaril Consultants}
\date{January 2016}
\maketitle
\begin{abstract}
This document presents the basic concepts of typesetting in a form usable by non-specialists. It is aimed at those who find themselves (willingly or unwillingly) asked to undertake work previously sent out to a professional printer, and who are concerned that the quality of work (and thus their corporate aesthetic) does not suffer.
\end{abstract}
\end{document}

In both cases the Abstract or Summary is OPTIONAL (that is, \LaTeX doesn’t force you to have one), but it’s rare to omit it because readers want and expect it, and it’s used by web indexing engines to let people find it. In practice, of course, you go back and type the Abstract or Summary after having written the rest of the document, but for the sake of the example we’ll jump the gun and type it now.

You add the \abstract environment after the \maketitle command, and type your Abstract or Summary there, leaving a blank
2.5 A little think about structure

It’s very easy to sit down at a keyboard with a traditional wordprocessor and just start typing. If it’s a very short document, or something transient or relatively unimportant, then you just want to type it in and make it ‘look right’ by highlighting with the mouse and clicking on font styles and sizes.
In doing so, you may achieve the effect you wanted, but your actions have left no trace behind of why you made these changes. This is usually unimportant for trivial or short-term documents, but if you write longer or more complex documents, or if you often write documents to a regular pattern, then making them consistent by manual methods becomes a nightmare. \LaTeX’s facilities for automation are based on you providing this ‘why’ information.

If your documents have any of the features below, then you have probably already started thinking about structure.

- The document naturally divides into sections (parts, chapters, etc).
- The document is long.
- There is lots of repetitive formatting in the document.
- The document is complex (intellectually or visually).
- There are lots of figures or tables (or examples, exercises, panels, sidebars, etc).
- Accuracy is important in formatting the document.
- A master copy is needed for future reference or reprinting.
- This is a formal or official document needing special care and attention.
- It’s my thesis, book, leaflet, pamphlet, paper, article, etc. That’s why I care.
- The document (or part of it) may need ongoing or occasional re-editing and republishing.

If you’ve got that far, you’re over half-way done. Using a structural editor — even a simple outliner — can make a huge difference to the quality of your thinking because you are consciously organising your thoughts before setting them down. And it can make just as big a difference to your formatting as well: more consistent, better presented, easier for the reader to navigate through, and more likely to be read and understood — which is presumably why you are writing the document in the first place.
2.6 Sections

\LaTeX provides seven levels of division or sectioning for you to use in structuring your text. They are all optional: it is perfectly possible to write a document consisting solely of paragraphs of unstructured text. But even novels are normally divided into chapters, although short stories are often made up just of paragraphs.

Table 2.1: \LaTeX's sectioning commands

<table>
<thead>
<tr>
<th>Depth</th>
<th>Division</th>
<th>Command</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>Part</td>
<td><code>\part</code></td>
<td>Not in letters</td>
</tr>
<tr>
<td>0</td>
<td>Chapter</td>
<td><code>\chapter</code></td>
<td>Books, reports</td>
</tr>
<tr>
<td>1</td>
<td>Section</td>
<td><code>\section</code></td>
<td>Not in letters</td>
</tr>
<tr>
<td>2</td>
<td>Subsection</td>
<td><code>\subsection</code></td>
<td>Not in letters</td>
</tr>
<tr>
<td>3</td>
<td>Subsubsection</td>
<td><code>\subsubsection</code></td>
<td>Not in letters</td>
</tr>
<tr>
<td>4</td>
<td>Titled paragraph</td>
<td><code>\paragraph</code></td>
<td>Not in letters</td>
</tr>
<tr>
<td>5</td>
<td>Titled subparagraph</td>
<td><code>\subparagraph</code></td>
<td>Not in letters</td>
</tr>
</tbody>
</table>

Chapters are only available in the book and report document classes, because they don’t have any meaning in articles and letters. Parts are also undefined in letters.\footnote{It is arguable that chapters also have no place in reports, either, as these are conventionally divided into sections as the top-level division. \LaTeX, however, assumes your reports have chapters, but this is only the default, and can be changed very simply (see section 7.6 on page 170).}

In each case the title of the part, chapter, section, etc goes in curly braces after the command. \LaTeX automatically calculates the correct numbering and prints the title in bold. You can turn section numbering off at a specific depth: details in section 2.6.1 on the following page.

There are packages to let you control the typeface, style, spacing, and appearance of section headings: it’s much easier to use them than to try and reprogram the headings manually. Two of the most popular are section and sectsty.

\begin{verbatim}
\section{New recruitment policies}
\subsection{Effect on staff turnover}
\end{verbatim}
\chapter{Business plan 2010-2020}

Headings also get put automatically into the Table of Contents, if you specify one (it’s optional). But if you make manual styling changes to your heading, for example a very long title, or some special line-breaks or unusual font-play, this would appear in the Table of Contents as well, which you almost certainly don’t want. \LaTeX{} allows you to give an optional extra version of the heading text which only gets used in the Table of Contents and any running heads, if they are in effect (see section 6.1.2 on page 135). This alternative heading goes in \emph{square brackets} before the curly braces:

\section{Effect on staff turnover}{An analysis of the effects of the revised corporate recruitment policies on staff turnover at divisional headquarters}

**Exercise 5. Start your document text**

1. Add a \chapter{} command after your Abstract or Summary, giving the title of your first chapter.

2. If you’re planning ahead, add a few more \chapter{} commands for subsequent chapters. Leave a few blank lines between them to make it easier to add paragraphs of text later.

3. Typeset the document.

### 2.6.1 Section numbering

All document divisions get numbered automatically. Parts get Roman numerals (Part I, Part II, etc); chapters and sections get decimal numbering like this document, and Appendices (which are just a special case of chapters, and share the same structure) are lettered (A, B, C, etc). You can easily change this default if you want some special scheme.

You can change the depth to which section numbering occurs, so you can turn it off selectively. In this document the depth is set to 3, using the depth column in Table 2.1 on the previous page. If you only want
parts, chapters, and sections numbered, not subsections, subsubsections, or lower levels, you can change the value of the \texttt{secnumdepth} counter using the the \texttt{\setcounter} command, giving the depth value from Table 2.1 on page 47:

\begin{verbatim}
\setcounter{secnumdepth}{1}
\end{verbatim}

Notice that the \texttt{\setcounter} command, like \texttt{\renewcommand} which we saw earlier, has two arguments: the name of the counter you want to set, and the number you want to set it to.

A related counter is \texttt{tocdepth}, which specifies what depth to take the Table of Contents to. It can be reset independently, in exactly the same way as \texttt{secnumdepth}. The setting for this document is 2.

\begin{verbatim}
\setcounter{tocdepth}{3}
\end{verbatim}

To get a one-time (special case) \texttt{unnumbered} section heading which does \texttt{not} go into the Table of Contents, follow the command name with an asterisk before the opening curly brace:

\begin{verbatim}
\subsection*{Shopping List}
\end{verbatim}

All the divisional commands from \texttt{part*} to \texttt{subparagraph*} have this ‘starred’ version which can be used in isolated circumstances for an unnumbered heading when the setting of \texttt{secnumdepth} would normally mean it would be numbered.

## 2.7 Ordinary paragraphs

After section headings comes your text. Just type it and leave a blank line between paragraphs. That’s all \LaTeX{} needs.

The blank line means ‘end the current paragraph here’: it does \texttt{not} (repeat: \texttt{not}) necessarily mean you get a blank line in the typeset output.

The spacing between paragraphs is an independently definable quantity, a \texttt{dimension} or \texttt{length} called \texttt{\setlength}. This is normally zero (no space between paragraphs, because that’s how books are normally typeset), but you can easily set it to any size you want with the command \texttt{\setlength} in your Preamble: like \texttt{\setcounter}
it takes two arguments: the name of the length, and the value to set it to:

\setlength{\parskip}{1cm}

This will set the space between paragraphs to 1cm. See section 1.9.1 on page 23 for details of the various size units \LaTeX can use. *Leaving multiple blank lines between paragraphs in your source document does not create extra space:* all extra blank lines are ignored by \LaTeX: the space between paragraphs is controlled *only* by the value of \parskip. White-space in \LaTeX can also be made flexible (what Lamport calls ‘rubber’ lengths). This means that values such as \parskip can have a default dimension plus an amount of expansion minus an amount of contraction. This is useful on pages in complex documents where not every page may be an exact number of fixed-height lines long, so some give-and-take in vertical space is useful. You can specify this in a \setlength command:

\setlength{\parskip}{1cm plus4mm minus3mm}

Paragraph indentation can also be set with the \setlength command, although you would always make it a fixed size, never a flexible one, otherwise you would have very ragged-looking paragraphs.

\setlength{\parindent}{6mm}

By default, the first paragraph after a chapter or section heading follows the standard Anglo-American publishers’ practice of no indentation. Subsequent paragraphs are indented by the value of \parindent (default 18pt).\footnote{Paragraph spacing and indentation are cultural settings. If you are typesetting in a language other than English, you should use the \texttt{babel} package, which alters many things, including the spacing and the naming of sections, to conform with the standards of different countries and languages.} You can change this in the same way as any other length.

In the printed version of this document, the paragraph indentation is set to 12.0pt and the space between paragraphs is set to 0.0pt plus 1.0pt. These values do not apply in the Web (HTML) version because not all browsers are capable of that fine a level of control, and
because users can apply their own stylesheets regardless of what this document proposes.

Exercise 6. Start typing!

1. Type some paragraphs of text. Leave a blank line between each. Don’t bother about line-wrapping or formatting — \LaTeX will take care of all that.

2. If you’re feeling adventurous, add a \texttt{section} command with the title of a section within your first chapter, and continue typing text below that.

3. Add one or more \texttt{setlength} commands to your Preamble to experiment with changing paragraph spacing and indentation.

To turn off indentation completely, set it to zero (but you still have to provide units: it’s still a measure!).

\texttt{\setlength{\parindent}{0in}}

If you do this, though, and leave \texttt{parskip} set to zero, your readers won’t be able to tell easily where each paragraph begins! If you want to use the popular office-document style of having no indentation with a space between paragraphs, use the \texttt{parskip} package, which does it for you (and makes adjustments to the spacing of lists and other structures which use paragraph spacing, so they don’t get too far apart).

2.8 Table of contents

All auto-numbered headings get entered in the Table of Contents (ToC) automatically. You don’t have to print a ToC, but if you want to, just add the command \texttt{tableofcontents} at the point where you want it printed (usually after the Abstract or Summary).

Entries for the ToC are recorded each time you typeset your document, and reproduced the next time you typeset it, so you need to re-run \LaTeX one extra time to ensure that all ToC page-number references are correctly resolved.
CHAPTER 2. BASIC STRUCTURES

The commands \texttt{listoffigures} and \texttt{listoftables} work in exactly the same way as \texttt{tableofcontents} to automatically list all your tables and figures. If you use them, they normally go after the \texttt{tableofcontents} command.

We’ve already seen in section 2.6 on page 47 how to use the optional argument to the sectioning commands to add text to the ToC which is slightly different from the one printed in the body of the document. It is also possible to add extra lines to the ToC, to force extra or unnumbered section headings to be included.

Exercise 7. Using a Table of Contents

1. Add the \texttt{tableofcontents} command to your document, after the \texttt{maketitle} command but before the Abstract.
2. Typeset the document.
   If you are using your editor’s Build to typeset the document, it should re-run \LaTeX if necessary, to make sure the Table of Contents is updated to reflect any changes you have made (for example in section numbering, or adding, deleting, or moving sections or chapters around.
3. If the Table of Contents does not reflect your document structure, you need to typeset it one more time, to bring it up to date.

The \texttt{tableofcontents} command normally shows only numbered section headings, and only down to the level defined by the \texttt{tocdepth} counter (see section 2.6.1 on page 48), but you can add extra entries with the \texttt{addcontentsline} command. For example if you use an unnumbered section heading command to start a preliminary piece of text like a Foreword or Preface, you can write:

\begin{verbatim}
\subsection*{Preface}
\addcontentsline{toc}{subsection}{Preface}
\end{verbatim}

This will format an unnumbered ToC entry for ‘Preface’ in the ‘subsection’ style. You can use the same mechanism to add lines to the List of Figures or List of Tables by substituting \texttt{lof} or \texttt{lot} for \texttt{toc}.  

Format}
2.8. TABLE OF CONTENTS

There is also a command \texttt{addtocontents} which lets you add any \LaTeX{} commands to the ToC file. For example, to add a horizontal rule and a 6pt gap, you could say

\texttt{\addtocontents{toc}{\par\hrule\vspace{6pt}}}

at the place where you want it to occur. You should probably only use this command once you know what you are doing.

There are several packages to help you restyle these lists of contents; perhaps the best-known is tocloft.
3 Packages and CTAN

The Comprehensive \TeX{} Archive Network (CTAN) is a repository of Web documents and files from HTTP and FTP servers worldwide which contain copies of almost every piece of free software related to \TeX{} and \LaTeX{}.

Always try CTAN first

CTAN should always be your first port of call when looking for a software update or a feature you want to use. To help prevent overload on the volunteers, please don’t ask the network help resources until you have checked CTAN and the FAQ (section 3.3.3 on page 67) first.

CTAN is based on three main servers, and there are several online indexes available. There are complete \TeX{} and \LaTeX{} systems for all platforms, utilities for text and graphics processing, conversion programs into and out of \LaTeX{}, printer drivers, extra typefaces, and (possibly the most important) the \LaTeX{} packages. The three main servers are:

- \TeX{} Users Group:  \url{www.ctan.org/}
- UK \TeX{} Users Group:  \url{www.tex.ac.uk/}
- Deutschsprachige Anwendervereinigung \TeX{} e.V. (DANTE, the German-speaking \TeX{} Users Group):  \url{dante.ctan.org/}
In addition, there are dozens of mirrors of CTAN around the world, run by ISPs, companies, universities, and other institutions. These automatically update themselves from the three main servers, so for a speedy download from your closest server, you can go to mirrors.ctan.org and it will redirect you to your nearest mirror site.

3.1 Packages and classes

Add-on (plug-in) features for \LaTeX{} are known as packages. Most of them can be used with any document class, to add extra formatting features. There over 5,000 packages, and several hundred of them are pre-installed with a full distribution of \LaTeX{} and can be used in your documents immediately. They are all stored in subdirectories of your \TeX{} installation directory, named after each package. To find out what packages are available and what they do, you should use the CTAN search page which includes a link to Graham Williams' comprehensive package catalogue. There are two main types of package:

**Packages**: A \TeX{} package is a file or collection of files containing extra commands and programming which add new formatting features, or modify those already existing. Installed package files all end with \texttt{.sty} (they used to be called ‘style files’) and they may also contain ancillary files as well like fonts or configurations files.

**Classes**: A \TeX{} class is a special kind of package which provides formatting template features for a whole document. There are many of these, and several dozen of them are pre-installed with \TeX{}. Class files all end with \texttt{.cls} and additional classes can also be downloaded from CTAN.

When you try to typeset a document which requires a package which is not installed on your system, \TeX{} will warn you with an error message that it is missing (see section C.3.3.7 on page 254), and you can then download the package and install it using the instructions in section 3.2 on page 59.

However, many \TeX{} distributions can now catch this error, and offer to download and install the missing class or package for you right there.
and then, and then carry on typesetting. This feature, the \TeX\ Live Package Manager (\texttt{tlmgr}), is not yet available in all distributions of \TeX, so check your documentation to see if it is working in your version. This useful facility avoids you having to do manual package installation except for a few packages that are very old or do not conform to the TDS standard (see section 3.2.1 on page 60).

You can also download updates to packages you already have, both the ones that were installed along with your version of \TeX\ as well as ones you have added. Updates occur when a class or package author finds and fixes a bug, or adds a new feature. All package updates on CTAN are automatically announced on the Usenet newsgroup \texttt{comp.text.tex}. See section 3.2 on page 59 for details of how to install packages and updates.

There is no limit to the number of packages you can have installed on your computer (apart from disk space!), but there is probably a physical limit to the number that can be used inside any one \TeX\ document at the same time, although it depends on how big each package is. In practice there is no problem in having even a couple of dozen packages active (this document uses over 30).

### 3.1.1 Using an existing class

We’ve already seen how to do this in section 2.1.1 on page 34: it’s the document class name that you put in curly braces in the \texttt{documentclass} line at the start of a \TeX\ document.

All classes have \texttt{options} (we saw some in use in the Quick Start document). The class documentation will explain what they are for and how to use them.

### 3.1.2 Using an existing package

To use a package already installed on your system, put a \texttt{\usepackage} command in your document Preamble with the package name in curly braces, as we have already seen in earlier chapters. For example, to use the \texttt{xcolor} package, which lets you typeset in colours (I warned you this was coming!), you would type:

\begin{verbatim}
\documentclass[11pt,a4paper,oneside]{report}
\usepackage{xcolor}
\begin{document}
\end{verbatim}
This makes available a \texttt{color} command and many others, and several sets of predefined palettes of colours which you can specify using options.

You can include several package names in one \texttt{usepackage} command by separating the names with commas, and you can have more than one \texttt{usepackage} command.

If you use the package options, you must give the package its own separate \texttt{usepackage} command, like \texttt{geometry} and \texttt{xcolor} shown below:

```
\documentclass[11pt,a4paper,oneside]{report}
\usepackage{pslatex,palatino,avant,graphicx}
\usepackage[margin=2cm]{geometry}
\usepackage[svgnames]{xcolor}
\begin{document}
\title{\textcolor{Crimson}{Practical Typesetting}}
\author{\textcolor{StateBlue}{Peter Flynn}\textbackslash Silmaril Consultants}
\date{\textcolor{ForestGreen}{January 2016}}
\maketitle
\end{document}
```

(Incidentally, don’t actually do this: it’s a very crude and cumbersome way to do colours in titling. It’s fine for a one-time short document, but it will interfere with running heads if you use them; and if it’s for a repeatable style we’ll see in Chapter 7 starting on page 161 how it can be automated as part of the \texttt{maketitle} command and kept out of the author’s way.)

**Exercise 8. Add colour**

Use the \texttt{xcolor} package to add some colour to your document. Stick with primary colours for the moment.

Use the \texttt{geometry} package to change the margins.

Reprocess and print your document if you have a colour printer (monochrome printers should print it in shades of grey).
3.2. INSTALLING CLASSES AND PACKAGES

The `geometry` package has options to let you specify margins, page and paper sizes, header and footer depths, and a lot of other page-geometry dimensions. The `xcolor` package has options to let you specify which of several standard palettes of colours you want to use.

It’s really important to read the documentation for the package concerned to find out what can be done and how to do it: see section 3.1.3 up next.

3.1.3 Package documentation

To find out what commands a package provides (and thus how to use it), you need to read the documentation. The simplest way is to use your command window and type `$texdoc` followed by the package name. This will bring up the documentation in your PDF or DVI viewer. Alternatively, use your system’s file finder to look for the package name — it should turn up the package directory itself as well as the documentation directory — what you’re looking for is a PDF or DVI document.

If that doesn’t find it, in the `texmf/doc` directory of your installation there should be subdirectories full of `.dvi` and `.pdf` files, one for every package installed. These can be previewed or printed like any other DVI or PDF file (see section C.4.1 on page 255). If your installation procedure has not installed the documentation, the files can all be viewed or downloaded from the package’s page on CTAN — `www.ctan.org/pkg/` followed by the package name.

Before using a package, you should read the documentation carefully, especially the subsection usually called ‘User Interface’, which describes the commands the package makes available. You cannot just guess and hope it will work: you have to read it and find out.

See the next section for details of how to generate the documentation for additional packages you install yourself.

3.2 Installing classes and packages

If you’re using a system which has the `tlmgr` auto-installer, you don’t need this section except for very rare occasions. The `TeX` Live Package Manager can be used to find, download, and install packages without you needing to know anything about where they come from or where
Exercise 9. Read all about it

Find and view (or print) the documentation on the geometry package you used in section 3.1.2 on page 58. Investigate some of the other package documentation files in the directory.

they get put. The rest of this chapter is for people who want to do it by hand.

If you have to install a package manually, you can use the indexes on any CTAN server to find the package you need and the directory where it can be downloaded.

3.2.1 Downloading packages

Some packages are available as ZIP files in \TeX\ Directory Structure (TDS) format, which is faster and more convenient to use. Others are just plain zip files, or you can download all the needed files individually.

3.2.1.1 Downloading a TDS package zip file

If you go to a package’s CTAN page (http://ctan.org/pkg/\textit{name}) this will show the package details, and if the package is available in TDS format, there will be a link labelled ‘TDS archive’ with a file ending in \texttt{.tds.zip}.

Download this and unzip it straight into your personal \TeX\ directory, where \TeX\ will find it first, overriding any other version that might have been installed with your distribution. The correct place for your personal \TeX\ directory is described in section A.2 on page 217.

3.2.1.2 Downloading a non-TDS package zip file

If there is no TDS zip file, there will be a prominent link at bottom right labelled ‘Download the contents of this package in one zip archive’.

Download the zip file to a temporary directory. If you use Windows, create a folder like \texttt{Computer\System\Users\your\ name\ temp}...
3.2. Installing Classes and Packages

or just C:\tmp or C:\temp for this;¹ Mac and Linux systems already have a /tmp directory.

Unzip it, then move the files into the directories as shown in Table 3.1 on page 63.

3.2.1.3 Manual download

If there is no ZIP file at all, what you need to look for is almost always two files, one ending in .dtx and the other in .ins. The first is a DOCTEX file, which combines the package programs and their documentation in a single file. The second is the installation program (much smaller). You MUST always download both these files (and maybe others in the download folder) if there is no ZIP file.

3.2.1.4 Other package downloads

If neither the two files nor the package ZIP are there, it means one of two things:

- Either the package is part of a much larger bundle which you SHOULD NOT normally update yourself unless you are updating your entire \LaTeX{} system;²

- or it’s one of a few rare or unusual packages still supplied as a single hand-made .sty or .cls file originally written for the now obsolete \TeX{} 2.09,³ or perhaps by an author who has a doctrinal or philosophical objection to using DOCTEX.

3.2.2 Installing a class or package manually

There are four steps to installing a \LaTeX{} class or package:

¹ MiKTeX users MUST note that you cannot process .ins files inside MiKTeX’s own installation folders: you have to process them elsewhere first, hence the need for a temporary directory.

² For example, there is no separate xcolor.dtx and xcolor.ins for the xcolor package because it forms part of the graphics bundle, which is included with all \LaTeX{} systems anyway. Such packages change very rarely, as they form part of the kernel of \LaTeX{} and are very stable. You should never try to update these packages in isolation.

³ Almost all of these have been updated to work with \LaTeX{} 2e, so they should be installed as in step 3 on the next page, but there are a few remaining.
CHAPTER 3. PACKAGES AND CTAN

On Unix-based systems (including Mac OS X and GNU/Linux), that’s all you need to do. On Windows systems running MiKTeX, you MUST reindex your File Name Database (FNDB, see step 4 on page 64) before \LaTeX{} will be able to find the new files.

1. **Extract the class or package files**

   Use your directory browser of file manager (eg My Computer, Finder, Thunar, Dolphin, etc) to find the folder where you unzipped or downloaded the .dtx and .ins files.

   Run \LaTeX{} on the .ins file. That is, open the file in your editor and process it as if it were a \LaTeX{} document (which it is), or if you prefer, type \texttt{$latex$} followed by the .ins filename in a command window in the directory where the file is.

   This will extract all the files needed from the .dtx file (which is why you must have both of them present in the directory).

   If this is a non-TDS zip file, or individually-downloaded files, note down or print the names of the files created if there are a lot of them (read the log file if you want to see their names again).

2. **Create the documentation**

   Run PDF\LaTeX{} on the .dtx file twice. This will create a .pdf file of documentation explaining what the package is for and how to use it. Two passes through \LaTeX{} are needed in order to resolve any internal crossreferences in the text (a feature we’ll come onto later). If there is a \texttt{BibTeX} file of references, or if you need the Index, you will need to process \texttt{bibtex}, \texttt{biber}, \texttt{makeindex}, or other ancillary programs. I very strongly recommend doing this with the \texttt{Build} menu of your editor, or with the \texttt{latexmk} tool.

3. **Install the files**

   This step is not needed if you used a TDS zip file.
3.2. INSTALLING CLASSES AND PACKAGES

Move the files created in step 1 on the facing page from your temporary directory to the right subdirectories in your personal \TeX{} directory (see below). Always put the files in your 'personal' \TeX{} directory tree, a) to prevent your new package accidentally overwriting master files in the main \TeX{} directories; and b) to avoid your newly-installed files being overwritten when you next update your version of \TeX{}. Never, never, NEVER put files into your \TeX{} distribution's main directory tree. (If you are updating a shared system, however, you can put the files into the local (shared) \TeX{} directory tree.)

Table 3.1: Where to put files from packages

<table>
<thead>
<tr>
<th>Type</th>
<th>Subdirectory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.cls</td>
<td>tex/latex/classname</td>
<td>Document class file</td>
</tr>
<tr>
<td>.sty</td>
<td>tex/latex/packagename</td>
<td>Package (style) file</td>
</tr>
<tr>
<td>.bst</td>
<td>bibtex/bst/packagename</td>
<td>Bib\TeX{} style file</td>
</tr>
<tr>
<td>.dvi</td>
<td>doc/packagename</td>
<td>package documentation</td>
</tr>
<tr>
<td>.pdf</td>
<td>doc/packagename</td>
<td>package documentation</td>
</tr>
<tr>
<td>.fd</td>
<td>tex/latex/mnfss</td>
<td>Font Definition files for METAFONT fonts</td>
</tr>
<tr>
<td>.fd</td>
<td>tex/latex/psnfss</td>
<td>Font Definition files for PostScript Type 1 fonts</td>
</tr>
<tr>
<td>.fd</td>
<td>tex/latex/typeface</td>
<td>Font Definition files for other fonts</td>
</tr>
<tr>
<td>.mf</td>
<td>fonts/source/public/typeface</td>
<td>METAFONT font outline</td>
</tr>
<tr>
<td>.pfb</td>
<td>fonts/type1/ foundry/ typeface</td>
<td>PostScript Type 1 outline</td>
</tr>
<tr>
<td>.ttf</td>
<td>fonts/truetype/ foundry</td>
<td>TrueType font files</td>
</tr>
<tr>
<td>.otf</td>
<td>fonts/opentype/ foundry</td>
<td>OpenType font files</td>
</tr>
<tr>
<td>.afm</td>
<td>fonts/afm/ foundry/ typeface</td>
<td>Adobe Font Metrics</td>
</tr>
<tr>
<td>.tfm</td>
<td>fonts/tfm/ foundry/ typeface</td>
<td>\TeX{} Font Metrics</td>
</tr>
<tr>
<td>.vf</td>
<td>fonts/vf/ foundry/ typeface</td>
<td>\TeX{} virtual fonts</td>
</tr>
<tr>
<td>.png</td>
<td>tex/generic</td>
<td>PNG images</td>
</tr>
<tr>
<td>.jpg</td>
<td>tex/generic</td>
<td>JPG images</td>
</tr>
<tr>
<td>others</td>
<td>tex/latex/packagename</td>
<td>other types of file unless instructed otherwise</td>
</tr>
</tbody>
</table>

If there are configuration or other files, read the documentation to find out if there is a special or preferred location to move them to.

‘The right place’ sometimes causes confusion, especially if your \TeX{} installation is old or does not conform to the TDS. For a TDS-
conformant system, ‘the right place’ is your personal \TeX{} directory tree unless you are a systems manager updating a shared machine, in which case it’s the local \TeX{} directory tree. Your personal \TeX{} directory tree is in your home directory (folder):

\begin{itemize}
  \item Unix & GNU/Linux systems: \texttt{/texmf/}.
  \item Apple Macintosh OS X: \texttt{~/Library/texmf}.
  \item Windows systems: \texttt{Computer/username/texmf} (on obsolete Windows systems you can use \texttt{C:\texmf}).
\end{itemize}

Create this directory now if it does not already exist. You will need to create subdirectories within this directory: see Table 3.1 on the previous page.

Often there is just a .\texttt{sty} file to move but in the case of complex packages there may be more, and they belong in different locations. For example, new Bib\TeX{} packages or font packages will typically have several subdirectories of files to install. This is why it is important to create a subdirectory for the package within your personal \TeX{} directory, rather than dump the files into \texttt{misc} along with other unrelated stuff.

4. **Shared systems and MIKTeX: update your index**

On Unix & GNU/Linux systems (including Apple Macintosh OS X) you MUST NOT run the \TeX{} indexer program or create an \texttt{ls-R} database in your personal \TeX{} directory. These systems search your personal \TeX{} directory automatically.

Otherwise:

\begin{itemize}
  \item Windows MIK\TeX{} users (only) MUST use the MIK\TeX{} Administration program to add your new personal \TeX{} folder to MIK\TeX{}’s search tree when you first create it.

  After that, each time you update files in there, you MUST run the File Name DataBase (FNDB) updater in the MIK\TeX{} Administration program, otherwise \TeX{} will never see your newly-installed files.

  \item If you are updating a shared system, putting the files into the local \TeX{} directory tree, you MUST run your \TeX{} indexer program afterwards to update the package database.
\end{itemize}
This program comes with every modern version of TeX and is variously called texhash, mktexlsr, or even configure, or it might just be a mouse click on a button or menu in your configuration system (like MIKTeX’s). Read the documentation that came with your installation to find out which it is.

On MIKTeX and shared systems, run your TeX indexer program after making changes

This step is essential, otherwise nothing will work.

Exercise 10. Install a package

Download and install the latest version of the enumitem package (which implements inline lists, among many other pieces of list formatting).

The tlmgr auto-updater is widely used in TeX Live systems except where TeX has been installed from Debian-based Unix system packages. On Windows and Apple Mac, and on Unix systems where TeX Live has been installed from the TUG DVD or download, tlmgr is the normal way to update packages. The manual process described above is only for those cases where tlmgr cannot be used.

This includes the thousands of installations which do not conform to the TDS, such as old shared Unix systems and some Microsoft Windows systems, so there is no way for an installation program to guess where to put the files: you have to know this yourself. There are also systems where the owner, user, or installer has chosen not to follow the recommended TDS directory structure, or is unable to do so for policy or security reasons (such as a shared system where she cannot write to a locked disk or directory).

The reason for having the local texmf directory (usually called texmf-local or texmf.local) is to provide a place for local
modifications on a shared or managed system (such as a server) which will override but otherwise not interfere with the main \TeX{} installation directory tree. Your installation should already be configured to look in the personal and local directories first, so that any updates to standard packages will be found there before the copies in the main \texttt{texmf} tree. All modern \TeX{} installations do this, but if not, you can edit \texttt{texmf/web2c/texmf.cnf} (or on a shared system, ask your systems manager or support person to do so). There is an example in section A.5 on page 226.

3.2.3 Replicating the TDS

If you have a distribution which has installed an auto-updater like \texttt{tlnmgr} then you’ll probably never have to update a package manually anyway, so you won’t need this section unless you need to install something from outside the standard distribution such as a private, corporate, or commercial package or typeface.

The \TeX{} Directory Structure is documented at \texttt{www.tug.org/tds/}. I find it useful to make the subdirectory structure of your personal \TeX{} directory (\texttt{eg \texttt{texmf}}, see section A.2 on page 217) the same as that of the main installation \texttt{texmf} directory, so that I have all the main branches of the tree ready for future use. Examine the subdirectories of \texttt{texmf/tex/latex/} in your installation for examples. For additions to packages which came with your \LaTeX{} distribution, you can then use the same subdirectory name and position in your personal \TeX{} directory as the original used in the main \texttt{texmf/...} directory. \LaTeX{} will then always use the updated version.

If you want to recreate the entire subdirectory structure ready for use, you can do it under Unix & GNU/Linux systems (including Apple Macintosh OS X) with the commands below. This example uses the Ubuntu/Debian directory \texttt{/usr/local/share/texmf} rather than Mac\TeX{}'s \texttt{/usr/local/texlive/yyyy/texmf-dist} (replacing \texttt{yyyy} with the year of the Mac\TeX{} distribution) or Red Hat’s \texttt{/usr/share/texmf-local}, so modify the \texttt{$cd$} command appropriately, and on a Mac, use \texttt{~/Library/texmf/\{\}} in the second command:

```
cd /usr/local/share/texmf
find . -type d -exec mkdir -p ~/texmf/{} \\;
```
3.3 Where to go for help

The indexes and documentation files in your \TeX{} installation and on CTAN are the primary online resource for self-help on specific packages, and you should read these carefully before asking questions about packages.

3.3.1 Beginners start here

A very valuable list of Dos and Don'ts is maintained on StackExchange listing the most common mistakes that newcomers make. Once you've got started with \LaTeX{}, especially if you have learned it informally from colleagues, it's worth having a look at this just to make sure you avoid the easiest pitfalls.

3.3.2 The Minimal [Non-]Working Example or MWE

If you want to send an example of what you're trying to do to one of the forums, mailing lists, or newsgroups listed here, you MUST send an Minimal [Non-]Working Example (MWE). This is your \LaTeX{} document pared right down to the bare metal: remove all non-relevant packages, all non-relevant commands and formatting, and send ONLY the absolute bare minimum necessary to show what doesn't work. Unless you do this, you are wasting everyone's time, including your own.

There is an excellent article by Talbot at tug.ctan.org/info/dickimaw/dickimaw-minexample.pdf which explains the procedure in fine detail (Talbot, 2014).

And guess what? While doing this, you often find you discover for yourself what the problem was, saving you and thousands of others the trouble of working it out afresh!

3.3.3 The FAQ

For general queries you should read the Frequently-Asked Questions (FAQ) document so that you avoid wasting your time and
others’ by asking about things for which there is already an easily-accessible answer.

The FAQ is managed by the UK \TeX\ Users Group and can be found at www.tex.ac.uk/faq/.

3.3.4 StackExchange

The web site tex.stackexchange.com is a carefully-managed and well-structured question-and-answer site for \TeX\ and \LaTeX\X. You can vote answers up or down according to their quality or usefulness, but there are strict rules about how you ask questions, the same as for comp.text.tex below.

3.3.5 The \TeX\hax mailing list

Another support resource is the mailing list texhax@tug.org. Again, feel free to ask questions, but again, try to answer the question yourself first (and say what you’ve tried in your message).

3.3.6 Web sites

The \TeX\ Users Group, as well as most local user groups, maintains a web site (www.tug.org) with lots of information about various aspects of the \TeX\ system. See Appendix 4 starting on page 261 for information on joining TUG.

3.3.7 Usenet News

The Usenet newsgroup comp.text.tex is the principal forum for other questions and answers about \TeX\ and \LaTeX\, as well as the principal place where new CTAN packages are announced.

Feel free to ask questions, but please do not ask frequently-asked questions: read the FAQ instead. The people who answer the questions do so voluntarily, unpaid, and in their own time. It is also important that for specific queries you include a Minimal [Non-]Working Example — a very short whole \LaTeX\ file that others can download and typeset, to see exactly what your problem is.

There is a very detailed guide to how to get the best out of asking questions on Usenet at www.catb.org/esr/faqs/smart-questions.html#intro.
3.3. WHERE TO GO FOR HELP

To access Usenet news, type the following URI into your browser’s ‘Location’ or ‘Address’ window: comp.text.tex (if your browser doesn’t support Usenet news, install one of the many free newsreaders⁴ — see the list at en.wikipedia.org/wiki/List_of_Usenet_newsreaders).

3.3.8 Google \LaTeX{} list

There is a Google Groups mailing list for \LaTeX{} users at groups.google.com/group/latexusersgroup?hl=en.

3.3.9 Commercial support

If you need commercial levels of support, such as 24-hour phone contact, or macro-writing services, you can buy one of the several excellent commercial versions of \TeX{} listed in Table 1 on page xxix, or contact a consultancy which deals with \TeX{} (details on the TUG Web site and in issues of TUGboat).

---

⁴ Note that this means newsreaders for the Usenet News (NNTP) service. It does not mean syndication readers for RSS, which are a different thing entirely — these are unfortunately also sometimes referred to as ‘newsreaders’.

Formatting Information
It is perfectly possible to write whole documents using nothing but section headings and paragraphs. As mentioned in section 2.6 on page 47, novels, for example, usually consist just of chapters divided into paragraphs. However, it’s more common to need other features as well, especially if the document is technical\(^1\) in nature or complex in structure.

In Chapter 2 starting on page 33 we saw how to create a hierarchical document structure with chapters and sections and paragraphs; this chapter covers the other building-blocks which you need within your structure: lists, tables, figures (including images), boxes like sidebars and panels, block quotations, and verbatim text (computer program listings). In Chapter 5 starting on page 107 we will cover the textual tools that you need inside text: footnotes, marginal notes, cross-references, citations, indexes, and glossaries.

\(^1\) It’s worth pointing out that ‘technical’ doesn’t necessarily mean ‘computer technical’ or ‘engineering technical’, least of all ‘mathematical technical’: it just means it contains a lot of \(\epsilon\), Greek for specialist material or artistry. A literary analysis such as *La Textualisation de Madame Bovary* (on the marginal notes in the manuscripts of Flaubert’s novel) is every bit as technical in the literary or linguistic field as the maintenance manual for the Airbus 380 is in the aircraft engineering field.
CHAPTER 4. LISTS, TABLES, FIGURES

4.1 Lists

Lists are useful tools for arranging thoughts in a digestible format, usually a small piece of information at a time. There are four basic types of list, shown in Table 4.1.

Table 4.1: Types of list

<table>
<thead>
<tr>
<th>Type of List</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random or itemized lists</td>
<td>(sometimes called ‘arbitrary’ or ‘bulleted’ lists) where the order of items is unimportant. The items are often prefixed with a bullet or other symbol for clarity or decoration, but are sometimes simply left blank, looking like miniature paragraphs (when they are known as ‘simple’ or ‘trivial’ lists).</td>
</tr>
<tr>
<td>Enumerated or ordered lists</td>
<td>(sometimes called ‘sequential’ or ‘numbered’ lists) where the order of items is critical, such as sequences of instructions or rankings of importance. The enumeration can be numeric (Arabic or Roman), or lettered (uppercase or lowercase), and can be programmed to be hierarchical (1.a.viii, 2.3.6, etc).</td>
</tr>
<tr>
<td>Descriptive or labelled lists</td>
<td>(sometimes called ‘discussion’ lists), which are composed of subheadings or topic labels (usually unnumbered but typographically distinct), each followed by one or more indented paragraphs of discussion or explanation.</td>
</tr>
<tr>
<td>Inline lists</td>
<td>which are sequential in nature, just like enumerated lists, but are a) formatted within their paragraph; b) usually labelled with letters like this example; and c) often mutually inclusive or exclusive, with the final item prefixed by ‘and’ or ‘or’ respectively.</td>
</tr>
</tbody>
</table>

There are actually two other types, segmented lists and reference lists, but these are much rarer, and outside the scope of this document.

The structure of lists in \LaTeX is identical for each type, but with a different environment name. Lists are another example of this \LaTeX technique (environments), where a pair of matched commands surrounds some text which needs special treatment.

Within a list environment, list items are always identified by the command \item (followed by an item label in [square brackets] in the case of labelled lists). You don’t type the bullet or the number or the formatting, it’s all automated.
4.1.1 Itemized lists

To create an itemized list, use the `itemize` environment:

\begin{itemize}
\item Itemized lists usually have a bullet;
\item Long items use ‘hanging indentation’, whereby the text is wrapped with a margin which brings it clear of the bullet used in the first line of each item;
\item The bullet can be changed for any other symbol, for example from the `\textsf{bbding}` or `\textsf{pifont}` package.
\end{itemize}

- Itemized lists usually have a bullet;
- Long items use ‘hanging indentation’, whereby the text is wrapped with a margin which brings it clear of the bullet used in the first line of each item;
- The bullet can be changed for any other symbol, for example from the `bbding` or `pifont` package.

The default list bullet is the normal round, solid one (•), which is also available with the command `\textbullet` if you load the `textcomp` package. See section 7.6.1 on page 171 for details of how to change the settings for list item bullets.

Nested itemized lists (see section 4.1.6 on page 77) used differing symbols for their bullets as well as more indentation and less spacing.

4.1.2 Enumerated lists

To create an enumerated list, use the `enumerate` environment:
CHAPTER 4. LISTS, TABLES, FIGURES

\begin{enumerate}
\item Enumerated lists use numbering on each item (can also be letters or roman numerals);
\item Long items use ‘hanging indentation’ in just the same way that itemized lists do;
\item The numbering system can be changed for any level.
\end{enumerate}

1. Enumerated lists use numbering on each item (can also be letters or roman numerals);

2. Long items use ‘hanging indentation’, in just the same way that itemized lists do;

3. The numbering system can be changed for any level.

See section 4.1.6 on page 77 for details of how to change the numbering schemes for each level.

In standard \LaTeX\ document classes, the vertical spacing between items, and above and below the lists as a whole, is more than between paragraphs. If you want tightly-packed lists, use the \texttt{enumitem} package, which provides an environment option \texttt{nosep} for the three main list environments (there is also a \texttt{nosep} option for even more compact spacing). Both these options come \texttt{after} the environment name, not before; eg \texttt{\begin{itemize}[nosep]}
4.1 Lists

\begin{description}
\item[Identification:] description lists require a topic for each item given in square brackets;
\item[Hanging indentation:] Long items use this in the same way as all other lists;
\item[Reformatting:] Long topic labels can be reprogrammed to fold onto multiple lines.
\end{description}

I very strongly recommend using the enumitem package with its \texttt{unboxed} environment option for description lists, which avoids the spacing problems with \LaTeX's default handling of long labels. This package has so many good features I tend to load it for virtually every document I create.

All three of these types of lists can have multiple paragraphs per item: just type the additional paragraphs in the normal way, with a blank line between each. So long as they are still contained within the enclosing environment, they will automatically be indented to follow underneath their item.

4.1.4 Inline lists

Inline lists are a special case, as they require the use of the enumitem or paralist packages.

The enumitem package with the \texttt{inline} option provides ‘starred’ versions of the three standard list types to do this: \texttt{enumerate*}, \texttt{itemize*}, and \texttt{description*}. It uses a specification in the optional argument for formatting the labels (for example, italic letters and an upright parenthesis), and it also provides extensive support for the punctuation and conjunction between items, making it unnecessary to type it separately for each item (and differently for the last-but-one).
\usepackage[inline]{enumitem}
...
\begin{itemize}
\item Inline lists, which are sequential in nature, just like enumerated lists, but are
\item formatted within their paragraph
\item usually labelled with letters
\item usually have the final item prefixed with ‘and’ or ‘or’, like this.
\end{itemize}

Inline lists, which are sequential in nature, just like enumerated lists, but are 
a) formatted within their paragraph; 
b) usually labelled with letters; and 
c) usually have the final item prefixed with ‘and’ or ‘or’, like this.

The paralist package provides an \texttt{inparaenum} environment, again with the optional formatting specification in square brackets:

\begin{itemize}
\item Inline lists, which are sequential in nature, just like enumerated lists, but are
\item formatted within their paragraph
\item usually labelled with letters; and
\item usually have the final item prefixed with ‘and’ or ‘or’, like this.
\end{itemize}

Inline lists, which are sequential in nature, just like enumerated lists, but are a) formatted within their paragraph; 
b) usually labelled with letters; and c) usually have the final item prefixed with ‘and’ or ‘or’, like this.

See Chapter 6 starting on page 129 for details of the font-changing commands used in the optional arguments to the \texttt{enumerate*} and \texttt{inparaenum} shown in these examples.
4.1. Lists

Exercise 11. List practice

Add some lists to your document. Pick any two of the ones described here to practice with.

Try the \texttt{enumitem} package or the \texttt{paralist} package (read the documentation).

4.1.5 Reference lists and segmented lists

Reference lists are visually indistinguishable from numbered or lettered lists, but the numbering or lettering does not imply a sequence. The numbers or letters are just used as labels so that the items can be referred to from elsewhere in the text (as in ‘see item 501(c)3’). In this sense they are really a kind of sub-sectional division, and \TeX’s \texttt{paragraph} or \texttt{subparagraph} commands (with appropriate renumbering) would probably be a far better solution than using a list. Label them and refer to them with \texttt{label} and \texttt{ref} as for any other cross-reference (see section 5.3 on page 109).

Segmented lists are a highly specialised structure and outside the scope of this document. For details of their usage, see the ‘TEI Guidelines’ (Burnard & Sperberg-McQueen, 1995).

4.1.6 Lists within lists

You can start a new list environment within the item of an existing list, so you can embed one list inside another up to four deep. The lists can be of any type, so you can have a description list containing an item in which there is a numbered sub-list, within which there is an item containing a bulleted sub-sub-list.

1. by default an outer enumerated list uses Arabic numerals;
   (a) an embedded enumerated list is lettered in lowercase;
   i. a third level is numbered in lowercase Roman numerals;
   A. the fourth level uses uppercase alphabetic letters.

Multiple embedded lists automatically change the bullet or numbering scheme so that the levels don’t get confused, and the spacing
CHAPTER 4. LISTS, TABLES, FIGURES

Table 4.2: Default numbering for nested numbered lists

<table>
<thead>
<tr>
<th>Level</th>
<th>Default</th>
<th>Counter</th>
<th>Label command</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>digit.</td>
<td>enumi</td>
<td>\theenumi</td>
</tr>
<tr>
<td>2</td>
<td>(letter)</td>
<td>enumii</td>
<td>\theenumii</td>
</tr>
<tr>
<td>3</td>
<td>roman.</td>
<td>enumiii</td>
<td>\theenumiii</td>
</tr>
<tr>
<td>4</td>
<td>LETTER.</td>
<td>enumiv</td>
<td>\theenumiv</td>
</tr>
</tbody>
</table>

between levels is adjusted to become slightly tighter for more deeply nested levels.

• by default the outer itemized list item has a bullet;
  – an embedded itemized list uses a dash;
    * a third level uses an asterisk;
    · the fourth level uses a small bullet.

These are only defaults and can easily be changed by redefining the relevant set of values. You could also add a fifth and further levels, although I suspect that would mean your document structure needed some careful analysis, as lists embedded five deep will probably confuse your readers.

The values for lists come in pairs: for each level there is a counter to count the items and a command to produce the label:²

Note that each counter and command ends with the Roman numeral value of its level (this is to overcome the rule that \LaTeX\ commands can only be made of letters — digits wouldn’t work here). To change the format of a numbered list item counter, just renew the meaning of its label:

\begin{verbatim}
\renewcommand{\theenumi}{\Alph{enumi}}
\renewcommand{\theenumii}{\roman{enumii}}
\renewcommand{\theenumiii}{\arabic{enumiii}}
\end{verbatim}

² In fact, any time you define a counter in \LaTeX, you automatically get a command to reproduce its value. So if you defined a new counter example to use in a teaching book, by saying \texttt{\newcounter{example}}, that automatically makes available the command \texttt{\theexample} for use when you want to display the current value of example.
This would make the outermost list use uppercase letters, the second level use lowercase roman, and the third level use ordinary Arabic numerals. The fourth level would remain unaffected.

### Exercise 12. Nesting

Extend your use of lists by nesting one type inside a different one.

## 4.2 Tables

Tabular typesetting is the most complex and time-consuming of all textual features to get right. This holds true whether you are typing in plaintext form, using a wordprocessor, using \LaTeX, using HTML or XML, using a DTP system, or some other text-handling package.

Fortunately, \LaTeX provides a table model with a mixture of defaults and configurability to let it produce very high quality tables with a minimum of effort.

### 4.2.1 Floats

Tables and Figures (and several other features of documents like sidebars) are what printers and publishers refer to as ‘floats’. This means they are not part of the normal stream of sentences of text, but separate freestanding entities, positioned in a part of the page to themselves (top, middle, bottom, left, right, or wherever the designer specifies). They

---

### Lists and Tables: a caution to the unwary

Treat lists with care: people sometimes use tables for labelled information which is really a list and would be better handled as such. They often do this because their wordprocessor has no way to do what they want (usually to place the item label level with the description or explanation) except by using a table, hence they are misled into believing that their text is really a table when it’s actually not.

---

*Formatting Information*
Terminology

\LaTeX, in common with standard typesetting practice, uses the word ‘Table’ to mean a formal textual feature, numbered, with a caption, and containing an aligned grid of numbers or text, referred to from the surrounding document (as in ‘See Table 5’). A Table is the whole thing, not just the grid, and — critically — it floats (see section 4.2.1 on the previous page).

The grid arrangement of information in rows and columns within either of these structures is called a ‘tabulation’ or ‘tabular matter’.

It is important to keep this distinction firmly in mind for this section.

You can also have ‘informal’ tables, which simply grids occurring between two paragraphs, without caption or number or reference: there’s one in section 1.4.1 on page 12.

always have a caption describing them and they are always numbered so they can be referred to from elsewhere in the text.

\LaTeX automatically floats Tables and Figures, depending on how much space is left on the page at the point that they are processed. If there is not enough room on the current page, the float is moved to the top of the next page. This can be changed by moving the Table or Figure to an earlier or later point in the text, or by adjusting some of the parameters which control automatic floating.

Authors sometimes want many floats occurring in rapid succession, which raises the problem of how they are going to fit on the page and still leave room for text. In this case, \LaTeX stacks them all up and prints them together if possible, or leaves them to the end of the chapter in protest.

The skill is to space them out within your text so that they intrude neither on the thread of your argument or discussion, nor on the visual balance of the typeset pages. But this is a skill few authors have, and it’s one point at which professional typographic advice or manual intervention may be needed.

There is a float package which lets you create new classes of floating object (sidebars, examples, exercises, etc), and it also implements a
method of forcing a float not to float (that is, to appear where it occurs in the text, even if that breaks the page layout).

Please now read from section 4.2.1 on page 79 up to here a second time.

### 4.2.2 Normal tables

To create a \LaTeX\ Table, use the `table` environment containing a `\caption` command followed by a `\label` command (the label can be used to refer to the table: see section 5.3.1 on page 110).

```
\begin{table}
  \caption{Project expenditure to year-end 2016}
  \label{ye2016exp}
  ...
\end{table}
```

Numbering is automatic, but the `\label` command \textit{MUST} follow the `\caption` command, not precede it. The numbering automatically includes the chapter number in document classes where this is appropriate (but this can of course be overridden). The `\caption` command has an optional argument to provide a short caption if the full caption would be too long for the List of Tables:

```
\caption[Something short]{Some very long caption that will only look reasonable in the full figure.}
```

### 4.2.3 Simple tabular matter

Within a Table, there are four ways to enter the data:

- **By hand**: you can enter the tabular matter (cell data) by typing it in, which is perhaps the most common method;

- **In a grid tool**: many \LaTeX\ editors come with a pop-up grid tool like a miniature spreadsheet, which makes creating tabular matter easier, at the cost of some loss of control;

- **With a package**: if the quantity of data is very large and is already in a spreadsheet, or if it is spreadsheet data which will change frequently before you are finished, you can use the `datatool`
package to read the data from a spreadsheet CSV export file (see section 4.2.4 on page 85);

**As an image**: it is also possible to include a ‘table’ which has actually been captured as an image from elsewhere, such as a screenshot from a spreadsheet (but then this isn’t really a table). We will see how to include images in section 4.3 on page 89 on Figures, where they are more common.

In Table 4.3 we have a table which we’ll use as an example. It’s got a number, a caption, three columns with headings and some ruled lines, and a comment afterwards.

<table>
<thead>
<tr>
<th>Item</th>
<th>€ Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>a) Salaries (2 part-time research assistants)</td>
<td>28,000</td>
</tr>
<tr>
<td>Conference fees and travel expenses</td>
<td>14,228</td>
</tr>
<tr>
<td>Computer equipment (5 workstations)</td>
<td>17,493</td>
</tr>
<tr>
<td>Software</td>
<td>3,562</td>
</tr>
<tr>
<td>b) Rent, light, heat, etc</td>
<td>1,500</td>
</tr>
<tr>
<td>Total</td>
<td>64,783</td>
</tr>
</tbody>
</table>

The Institute also contributes to (a) and (b).

To typeset tabular matter, within a table environment or elsewhere, you use the `tabular` environment. This takes one compulsory argument which specifies how many columns and what type they are. You follow the \begin{tabular} \end{tabular} command with a pair of curly braces giving the alignment of the columns.

**Column alignment**: You give one letter for each column using one of *l*, *c*, and *r* for a left-aligned, centered, or right-aligned column. The number of letters **MUST** be the same as the number of columns you are putting in the table.

\begin{tabular}{clr} ... \end{tabular}

In the example in Table 4.3, the tabular setting has three columns, the first one centered, the second left-aligned, and the third one right-aligned, so it is specified as `{clr}`.
4.2. TABLES

**Cell and row division**: You can then type in each row, making sure each cell’s data in the row is separated with an `&` character, and each row ends with a double backslash (`\`).

```
\begin{tabular}{l}
    a)& Salaries (2 part-time research assistants) & 28,000 \\
\end{tabular}
```

You don’t need to add any extra spaces or do any manual formatting, although you can if you want: \LaTeX{} just uses the column specification to know how to format it.

If a cell has nothing to go in it, you just don’t type anything, but the ampersand must still be there:

```
\begin{tabular}{l}
    & Total & 64,783 \\
\end{tabular}
```

**Column headings**: These are often set in **bold type**, as in the example (see ‘Cell formatting’ below).

```
\begin{tabular}{ll}
    \textbf{Item} & \textbf{EUR Amount} \\
\hline
\end{tabular}
```

In this case there is also some extra space (6pt, see ‘Row spacing’ below) and a horizontal line across the table (see ‘Tables rules’ below).

All the data for a row may be longer than a line in your editor, but it can take up as many lines as needed; the end of the row is signalled by the double backslash, so \LaTeX{} knows when it’s time for the next row.

**Cell formatting**: Font changes can be done within a cell (bold, italic, etc; we’ll come on to these later, see section 6.2.6 on page 149) and these changes are limited to the cell in which they occur: they do not ‘bleed’ across cells (in the example, the column headings have each been made bold separately).

**Row spacing**: Additional vertical white-space below a row (but above a rule) can be specified by giving a dimension in [square brackets] immediately after the double backslash which ends the row (3pt in the case of the last row before the totals in the example). A negative value will decrease the spacing below that row.
If the line below a horizontal rule looks too close, it can be optically spaced by adding a \texttt{strut} at the start of the next line (that is, \texttt{after} the \texttt{\hline}). A ‘strut’ is hidden vertical rule a little bit higher than the row-height; hidden because its width is zero, making it invisible, as in the example code. The \texttt{\rule} command can be used for this, with a width of 0pt and height of 1.2em, just a fraction higher than the text, forcing the lines apart by 0.2em.

\begin{table}
\caption{Project expenditure to year-end 2016}
\centering\smallskip
\begin{tabular}{clr}
\textbf{Item} & \textbf{EUR Amount} \\
\hline
a) & Salaries (2 \texttt{part-time research assistants})\&28,000 \\
& Conference fees and travel expenses\&14,228 \\
& Computer equipment (5 \texttt{workstations})\&17,493 \\
& Software\&3,562 \\
b) & Rent, light, heat, etc.\&1,500 \\
\hline
\texttt{\rule(0pt){1.2em}}
\end{tabular}
\par\medskip\footnotesize
The Institute also contributes to (a) and (b).
\end{table}

**Table rules:** A line across the whole table is done with the \texttt{\hline} command after the double-backslash which ends a row.

For a line which only covers some of the columns, use the \texttt{\cline} command (in the same place), with the column range to be ruled in curly braces. If only one column needs a rule, it must still be given as a range (eg in the example, \{3–3\}).

Vertical rules (between columns) can be specified in the column specifications with the vertical bar character (|) before, after, or between the \texttt{l}, \texttt{c}, \texttt{r} letters. This character creates rules which extend the whole height of a table: it is not necessary to repeat them every row.

I have indented the code example given just to make the elements of the table clearer to read: this is for editorial convenience, and has no effect on the formatted result (see Table 4.3 on page 82). If you copy and paste this into your example document, you will need to add the \texttt{marvosym} package to your Preamble, which will let you
4.2 TABLES

use the official CEC-conformant Euro symbol command $\texttt{\textbackslash EUR}$ (€ as distinct from €).

4.2.4 More complex tabular formatting

\TeX's original tabular settings were designed for classical numerical grids, where each cell contains a single value. If you need a cell to contain multiline text, like a miniature paragraph, you can use the letter \texttt{p} followed by a width in curly braces instead of an \texttt{l}, \texttt{c}, or \texttt{r}. Thus \texttt{p\{3.5\text{cm}\}} would mean a column in which each cell would be 3.5cm wide, and could contain more than one line of text or values.

\begin{tabular}{cp\{3.5\text{cm}\}r}

These \texttt{p} column specifications are \textit{not} multi-row (row-spanned) entries: they are single cells which contain multiple lines of typesetting: the distinction is extremely important. These paragraphic cells are typeset justified (two parallel margins) and the baseline of the top line of text is aligned with the baseline of neighbouring cells in the row.

The \texttt{array} package provides some important enhancements which overcome the limitations of the \texttt{p} cells:

**Vertical alignment**: In addition to the \texttt{p}, whose vertical alignment baseline is the the top line of text, the \texttt{array} package provides the \texttt{m} and \texttt{b} letters. These work the same way as \texttt{p} (followed by a width in curly braces), but their vertical alignment baseline is the middle or bottom of the cell respectively.

**Prefixes and suffixes**: With the \texttt{array} package, any column specification letter can be preceded by > and some \LaTeX commands in curly braces. These commands are applied to every cell in that column, so to make a \texttt{p} column typeset ragged-right you would say, for example, >\{	exttt{\raggedright}p\{3.5\text{cm}\}\} (or \texttt{\raggedleft}, or \texttt{\centering}).

Note that if you do this, the final column specification must include a prefix or suffix containing the \texttt{\arraybackslash} command, to revert the meaning of the double-backslash, which gets redefined by horizontal formatting commands like \texttt{\raggedright}.
There is a suffix format as well: you can follow a column letter with \textless{} and code in curly braces (often used to turn off math mode started in a prefix).

The \texttt{colortbl} package lets you colour rows, columns, and cells; and the \texttt{dcolumn} package provides decimal-aligned column specifications for scientific or financial tabulations. Multi-column (column-spanning) is built into \LaTeX{} tables with the \texttt{\textbackslash\multicolumn} command; but for multi-row (row-spanning) cells you need to add the \texttt{multirow} package. Multi-page and rotated (landscape format) tables can be done with the \texttt{longtable}, \texttt{rotating}, and \texttt{landscape} packages.

The \LaTeX{} table model is very different from the HTML auto-adjusting model used in web pages; it’s closer to the CALS model used in technical documentation. However, auto-adjusting column widths are possible with the \texttt{tabularx} and \texttt{tabulary} packages, offering different approaches to dynamic table formatting.

You do not need to format the tabular data in your editor: \LaTeX{} does this for you when it typesets the table, using the column specifications you provided. You can give the cell values all on one line, or split over many lines: it makes no difference so long as cells are separated with the \& and rows are ended with the double-backslash.

As mentioned earlier, some editors have a grid-like array editor for entering tabular data. Takaaki Ota provides an excellent \texttt{tables-mode} for Emacs which uses a spreadsheet-like interface and can generate \LaTeX{} table source code (see Figure 4.1 on the next page).

If your tabular data comes from outside \LaTeX{}, Nicola Talbot’s excellent \texttt{datatool} package allows the import of data from spreadsheet .csv files and other sources.

### 4.2.5 More on tabular spacing

Extra space, called a ‘shoulder’, is automatically added on both sides of all columns by default. The default value is 6pt, so you get that amount left and right of the tabulation; because it is added left and right of every cell, the space between columns is therefore 12pt by default. This can be adjusted by changing the value of the \texttt{\textbackslash\tabcolsep} dimension before you begin the tabular environment.

\begin{verbatim}
\setlength{\tabcolsep}{3pt}
\end{verbatim}
The shoulder can be omitted in specific locations by adding the code `{}` in the appropriate places. For example to omit it at the left-hand and right-hand sides of a tabular setting, put it at the start and end of the column specifications (putting it between two column specifications will remove all space between those columns).

\begin{tabular}{@{}clr@{}}
\end{tabular}

To change the row-spacing in a tabular setting, you can redefine the \texttt{\texttt{\texttt{arraystretch}}} command (using \texttt{\texttt{\texttt{renewcommand}}} because it’s defined as a command, not a length). \texttt{\texttt{\texttt{arraystretch}}} is actually a \texttt{\texttt{\texttt{multiplier}}}, preset to 1, so \texttt{\texttt{\texttt{\texttt{renewcommand}}}}\texttt{\texttt{\texttt{\texttt{\texttt{\texttt{arraystretch}}}}}\texttt{\texttt{\texttt{\texttt{\texttt{\texttt{\texttt{}}}}}1.5}}}} would make the lines of your tabular settings one and a half times bigger than normal.

It is conventional to centre the tabular setting within a Table, using the \texttt{\texttt{center}} environment (note US spelling) or the \texttt{\texttt{\texttt{centering}}} command (as in the example), but this is an æsthetic decision (or

4.2. TABLES

Figure 4.1: Tables mode for Emacs
perhaps one mandated by your publisher: some journals insist instead that all tabular material is set flush to the left-hand margin).

If there is no data for a cell, just don’t type anything — but you still need the & separating it from the next column’s data. The astute reader will already have deduced that for a table of \( n \) columns, there must always be \( n - 1 \) ampersands in each row. The exception to this is when the \texttt{multicolumn} command is used to create cells which span multiple columns, when the ampersands of the spanned columns are omitted.

### 4.2.6 Tabular techniques for alignment

As mentioned earlier, it’s also perfectly possible to typeset tabular matter outside a formal Table, where you want to lay out an informal tabulation where a fully floating formal Table would be unnecessary (these are usually quite short: there are several of them in this document).

By default, \LaTeX\ typesets \texttt{tabular} environments \textit{inline} to the surrounding text (that is, within the paragraph, as if the whole \texttt{tabular} environment was a character), so if you want your alignment displayed by itself, put it between paragraphs, inside a positioning environment like \texttt{center, flushright}, or \texttt{flushleft}, or leave a blank line or \texttt{par} before and after so it gets typeset separately.

One side-effect of this is that small and intricately constructed micro-tabulations can be used to good effect when creating special effects like logos, as they get treated like a character and can be typeset anywhere.

\begin{verbatim}
\begin{tabular}
  @{} \\
  >{|\raggedright\textwidth} \\
  @{} \\
  >{|\raggedleft\arraybackslash\textwidth} \\
  @{} \\
\end{tabular}

left-hand material & right-hand material\\
\end{verbatim}

Tabular setting can also be used wherever you need to align material side by side, such as in designing letterheads, where you may want
your company logo and address on one side and some other information
on the other side to line up with each other. One common way to
implement ‘spring’ margins like this is to create two columns each half
the width of the page, allowing for the extra space that would otherwise
be added automatically between columns and at the edges:

As mentioned earlier, the \@{} suppresses the inter-column gap (or
the shoulder left or right) so that the total width available will be the
full text width of the page.

**Exercise 13. Create a tabulation**

Create one of the following in your document:

1. a formal Table with a caption showing the number of people in
   your class broken down by age and sex;
2. an informal tabulation showing the price for three products;

### 4.3 Figures

As explained in section 4.2.1 on page 79, Figures and Tables float to a
vacant part of the page, as they are not part of your normal text, but
illustrative objects that you refer to.

\begin{figure}
\caption{Total variable overhead variance (after
\textcite[p.191]{bull})
\label{workeff}
\centering
\fbox{\includegraphics[width=.75\columnwidth]{diagram}}
\end{figure}

To create a figure, use the figure environment. Like Tables, they
automatically get numbered, and they MUST include a \caption (with
a \label after it, if needed for cross-referencing). Like Tables, it is
conventional to centre the material, but that is a personal choice.
You can see that the structure is very similar to the `table` environment, but in this case we have a graphic included with the `\includegraphics` command. Here, it's also enclosed in an `\fbox`, which places a frame box around it (see section 4.6.2 on page 101). Details of including graphics are in the next section: you need the `graphicx` package. Details of the bibliographic citation mechanism used in the caption are in section 5.3.2 on page 111.

Figures can contain text, diagrams, pictures, or any other kind of illustration, even a `tabular` environment — LaTeX is agnostic on this point, so Tables can contain an image (of a table, presumably) and Figures can contain a tabulation. What matters is that you describe them properly.

Figure 4.2: Total variable overhead variance (after Bull (1972, p. 191))

The content of the Figure could of course also be textual, in the form of lists, paragraphs, or other blocks of text. For drawings, LaTeX has a very simple drawing environment called `picture`, which lets you create a limited set of lines and curves, but for a diagram of any complexity, you should use a standard vector drawing program (see ??)
4.4 Images

Images (graphics) can be included anywhere in a \LaTeX document, although in most cases of formal documents they will occur in Figures (see preceding section). To use graphics, you need to use the `graphicx` package in your Preamble: `\usepackage{graphicx}`.\footnote{You may find a lot of old files which use a package called `epsf`. Don’t use it: It’s obsolete.} This package provides the command `\includegraphics` which is used to insert an image in a document. The command is followed by the name of your graphics file \textit{without the filetype} (we’ll see in a minute why you don’t normally need to include the filetype).

\begin{verbatim}
\includegraphics{myhouse}
\end{verbatim}

In most cases you should just make sure the image file is in the same folder (directory) as the document you use it in. This avoids a lot of messing around remembering where you put the files; but you could instead put them all in a single folder and include that as part of the filename you use in the command.

\begin{verbatim}
\includegraphics{images/myhouse}
\end{verbatim}

If you have images you want to use in several different documents in different places on your disk, there is a way to tell \LaTeX where to look (see section 4.4.4 on page 96).

\subsection{Supported image file formats}

The type of image file you use depends on \LaTeX processor you are using (see section 1.2 on page 4 for how to choose). The common file types are:
Joint Photographic Experts Group (JPG), used for photographs and scanned images;
Portable Network Graphic (PNG), used for photographs and scanned images;
Portable Document Format (PDF), used for vector graphics (drawings, diagrams) and typographic output from other programs;
Encapsulated Postscript (EPS), an old publishing industry standard for many years, and the forerunner of PDF, still used by some older programs that generate diagrammatic or typographic output.

See ?? on page ?? and ?? on page ?? for other file formats. For more details, see the answers to the question Which graphics formats can be included in documents processed by latex or pdflatex?

For $\LaTeX$ and $\text{PDFLaTeX}$ (creating PDF output): Graphics files MAY be in PNG, PDF, or JPG (JPEG) format.

For the original $\LaTeX$ (creating DVI output): Graphics MUST be in EPS format: no other format will work (see ?? on page ??).

1. PNG actually gets converted to the PDF internal format automatically (at a small penalty in terms of speed) so for lots of images, or very large images, use JPG format or preconvert them to PDF;

2. It is also of course possible to convert (repackage) your JPG pictures to PDF, using any of the standard graphics conversion/ manipulation programs (see ?? on page ?? for details). Preconverting all your images to PDF makes them load into your document slightly faster.

3. $\LaTeX$ and $\text{PDFLaTeX}$ will search for the graphic file by file type, in this order (check for the newest definition in your pdftex.def): .png, .pdf, .jpg, .mps, .jpeg, .jbig2, .jb2, .PNG, .PDF, .JPG, .JPEG, .JBIG2, and .JB2. Thanks to Enrico Gregorio and Philipp Stephani on comp.text.tex for locating this for me.

4. See ?? on page ?? for more about how to create and manage your image files.
4.4. IMAGES

sam2p

Péter Szabó’s `sam2p` utility converts from several image formats to PDF (or EPS). It is available precompiled for Windows and Linux; Linux users may also find it in their distribution’s repositories. Due to the way the program is compiled it is not included in the TeX Live distributions.

4.4.1 Other file formats

Convert them to one of the supported formats using a graphics editing or conversion tool such as GIMP or ImageMagick.

Some commercial distributions of TeX systems allow other formats to be used, such as GIF, Microsoft Bitmap (BMP), or Hewlett-Packard’s Printer Control Language (PCL) files, and others, by using additional conversion software provided by the supplier; but you cannot send such documents to other \TeX{} users and expect them to work if they don’t have the same distribution installed as you have.

It is in fact possible to tell \TeX{} to generate the right file format by itself during processing, but this requires an external command-line graphics converter, and as it gets done afresh each time, it may slow things down rather a lot.

4.4.2 Postscript

Since \TeX{} 2010, EPS files will be automatically converted to PDF in PDFLaTeX or XeLaTeX if you include the `epstopdf` package. This avoids need to keep your graphics in two formats, at the expense of a longer compile time while it converts every EPS image (not recommended).

All good graphics packages (e.g., GIMP, PhotoShop, Corel Draw, etc) can save images as EPS, but be very careful with other software such as statistics, engineering, mathematical, and numerical analysis packages, because some of them, especially on Microsoft Windows platforms, use a very poor quality driver, which in turn creates very poor quality EPS files. If in doubt, check with an expert. If you find an EPS graphic doesn’t print, the chances are it’s been badly made by the creating software. Downloading Adobe’s own Postscript driver from their Web site and using that instead may improve things, but the only real solution is to use software that creates decent output.
For these reasons, if you create vector EPS graphics, and convert them to PDF format, do not keep additional JPG or PNG copies of the same image in the same directory, because they risk being used first by PDFLaTeX instead of the PDF file, because of the order in which it searches.

EPS files, especially bitmaps, can be very large indeed, because they are stored in ASCII format. Staszek Wawrykiewicz has drawn my attention to a useful MS-DOS program to overcome this, called cep (‘Compressed Encapsulated PostScript’) available from CTAN archive in the support/pstools directory, which can compress EPS files to a fraction of their original size. The original file can be replaced by the new smaller version and still used directly with \includegraphics.

One final warning about using EPS files with \includegraphics: never try to specify an absolute path (one beginning with a slash) or one addressing a higher level of directory (one beginning with ../). The dvips driver will not accept these because they pose a security risk to PostScript documents. Unlike PDF, PostScript is a real programming language, capable of opening and deleting files, and the last thing you want is to create a document able to mess with your filesystem (or someone else’s).

### 4.4.2 Resizing images

The \includegraphics command can take optional arguments within square brackets before the filename to specify the height or width, as in the example below. This will resize the image that prints; whichever dimension you specify (height or width) the other dimension will automatically be scaled in proportion to preserve the aspect ratio.

The file on disk does not get changed in any way, and nor does the copy included inside the PDF: what gets changed is just the size that it displays at in the finished document. So if you include a huge JPG but tell \LaTeX to print it at a small size, your PDF will still include the whole image file at full size — all that changes is the way it shows it. This is very inefficient: normally you should create images at the right size for the document.
4.4. IMAGES

\begin{center}
\includegraphics[width=5cm]{twithcat}
\end{center}

If you specify both height and width, the image will be distorted to fit (not really useful except for special effects). You can scale an image by a factor (using the \textit{scaled} option) instead of specifying height or width; clip it to specified coordinates; or rotate it in either direction. Multiple optional arguments are separated with commas.

For details of all the arguments, see the documentation on the \texttt{graphicx} package or a copy of the \textit{Companion}. The package also includes commands to \texttt{rotate}, \texttt{mirror}, and \texttt{scale} text as well as images.

4.4.3 Making images

There are two types of image: bitmaps and vectors.

\textbf{Bitmaps}: Bitmap images are made of coloured dots, so if you enlarge them, they go jagged at the edges, and if you shrink them, they may go blurry. Bitmaps are fine for photographs, where every tiny dot is a different colour, and the eye won’t notice so long as you don’t shrink or enlarge too much. Bitmaps for diagrams and drawings, however, are almost always the wrong choice, and often disastrously bad.

\textbf{Vectors}: Vector drawings are made from instructions, just like \LaTeX{} is, but using a different language (eg ‘draw this from here to here, using a line this thick’). They can be enlarged or reduced as much as you like, and never lose accuracy, because they get redrawn.
automatically at any size. You can’t do photographs as vectors, but vectors are the only acceptable method for drawings or diagrams.

Figure 4.3: The vector diagram from Figure 4.2 on page 90 shrunk and enlarged

Vector graphic packages are also better suited for saving your image directly in EPS or PDF format (both of which use their own vector language internally). All the major graphics-generating packages in all disciplines output vector formats: AutoCAD, ChemDraw, MathCAD,
4.4. IMAGES

Maple, Mathematica, ArcInfo, and so on. EPS was for decades the universally-accepted format for creating vector graphics for publication, with PDF a close second. PDF is now the most common format, but most of the major graphics (drawing) packages can still save as EPS, such as Photoshop, PaintShop Pro, Adobe Illustrator, Corel Draw, and GIMP. There are also some free vector plotting and diagramming packages available like InkScape, tkPaint, and GNUplot which do the same. Never, ever (except in the direst necessity) create any diagram as a bitmap.

Bitmap formats like JPG and PNG are ideal for photographs, as they are also able to compress the data substantially without too much loss of quality. However, compressed formats are bad for screenshots, if you are documenting computer tasks, because too much compression makes them blurry. The popular Graphics Interchange Format (GIF) is good for screenshots, but is not supported by \LaTeX: use PNG instead, with the compression turned down to minimum. Avoid uncompressible formats like BMP as they produce enormous and unmanageable files. The Tagged Image File Format (TIFF), popular with graphic designers, should also be avoided if possible, partly because it is even vaster than BMP, and partly because far too many companies have designed and implemented non-standard, conflicting, proprietary extensions to the format, making it virtually useless for transfer between different types of computers (except in faxes, where it’s still used in a much stricter version).

4.4.4 Graphics storage

I mentioned earlier that there was a way to tell \LaTeX where to look if you had stored images centrally for use in many different documents. The answer is in a command \texttt{\graphicspath} which you supply with an argument giving one or more names of additional directories you want searched when a file uses the \texttt{\ includegraphics} command.

Put the path in an additional set of curly braces (this lets you add more paths later: each in their own subset of curly braces). I’ve used the ‘safe’ (MS-DOS) form of the Windows My Pictures folder in the example because you should never use directory or file names containing spaces (see the panel ‘Picking suitable filenames’ on p. 39).
**Exercise 14. Adding pictures**

Add `\usepackage{graphicx}` to the Preamble of your document, and copy or download an image you want to include. Make sure it is a JPG, PNG, or PDF image if you use XeLaTeX or PDFLaTeX; or an EPS image if you use standard LaTeX.

Add `\includegraphics` and the filename in curly braces (without the filetype), and process the document and preview or print it.

Make it into a figure following the example in section 4.3 on page 89.

Be aware that some DVI previewers are not able to display all types of graphics, and some cannot display colour. For best results, use PDF or PostScript previewers.

Using `\graphicspath` does make your file less portable, though, because file paths tend to be specific both to an operating system and to your computer, like the examples above.

```
\graphicspath{{c:/mypict-1/camera}{z:/corp/imagelib}}
\graphicspath{{/var/lib/images}{/home/peter/Pictures}}
```

If you use original \LaTeX and `dvips` to print or create PostScript files, be aware that some versions will not by default handle EPS files which are outside the current directory, and will issue the error message saying that it is ‘unable to find’ the image. As we mentioned above, this is because PostScript is a programming language, and it would theoretically be possible for a maliciously-made image to contain code which might compromise your system. The decision to restrict operation in this way has been widely criticised, but it seems unlikely to be changed. If you are certain that your EPS files are kosher, use the `R0` option in your command, eg `$dvips -R0 ... dvifile$`

### 4.5 Quotations

Direct speech and short quotes within a sentence ‘like this’ are done with simple quotation marks as described in section 1.7 on page 17. Sometimes, however, you may want longer quotations set as a separate
4.5. QUOTATIONS

paragraph. Typically these are indented from the surrounding text. \LaTeX{} has two environments for doing this.

Such quotations are often set in a smaller size of type, although this is not the default, but you can use one of the size commands like \texttt{\small} (see section 6.2.7 on page 151) as shown in the example.

\begin{quotation}\small\noindent
At the turn of the century William Davy, a Devonshire parson, finding errors in the first edition of his \citetitle{davy}, asked for a new edition to be printed. His publisher refused and Davy purchased a press, type, and paper. He harnessed his gardener to the press and apprenticed his housemaid to the typesetting. After twelve \textquotesingle{}years work, a new edition of fourteen sets of twenty-six volumes was issued---which surely indicates that, when typomania is coupled with religious fervour, anything up to a miracle may be achieved.\hfill\textcite[p.76]{ryder}
\end{quotation}

\begin{quote}
At the turn of the century William Davy, a Devonshire parson, finding errors in the first edition of his \textit{A System of Divinity}, asked for a new edition to be printed. His publisher refused and Davy purchased a press, type, and paper. He harnessed his gardener to the press and apprenticed his housemaid to the typesetting. After twelve years work, a new edition of fourteen sets of twenty-six volumes was issued---which surely indicates that, when typomania is coupled with religious fervour, anything up to a miracle may be achieved.

(\textsc{Ryder}, 1976, p. 76)
\end{quote}

The inclusion of a bibliographic citation at the end is optional but commonplace, especially in academic or research documents where it may be compulsory because of the need to cite everything you quote. It’s also possible for this to be tucked into the space at the end of the last line of the quotation, if there is room (if it’s too long, it obviously has to go on a line by itself).

The \texttt{quotation} environment sets the whole block of text indented, and each paragraph of it also has its own indentation on the first line, even the first paragraph. This is rather unconventional as a default, so it is common to add a \texttt{\noindent} command at the start.
of the quotation so that the first paragraph does not get indented (others still will).

4.6 Boxes, sidebars, and panels

\LaTeX, like most typesetting systems, works by setting text into boxes. Each character is a box, with a height and a width, just like it is in metal type; characters are assembled into lines, which are also boxes; and lines are assembled into pages, which are also boxes. The page-making mechanism also works like an old compositor’s galley (tray) from the days of metal type: the box accumulates lines of typeset text until it’s a bit longer than the height of the page. \TeX then works out how much of it really will fit on the page, cuts it off and ships it out to the DVI or PDF file, and puts the remainder back into the galley (box) at the top, ready to start accumulating more material for the following page.

4.6.1 Boxes of text

Because of this ‘box model’, \LaTeX can typeset any text into a box of any width. The simplest command for small amounts of text is \texttt{\parbox}. This command needs two arguments in curly braces: the first is the width you want the text set to, and the second is the text itself, as in the example shown.

\texttt{\parbox{3in}{Please make sure you send in your completed forms by January 1st next year, or the penalty clause 2(a) will apply.}}

The text is typeset to the required width, and the box is extended downwards for as long as is required to fit the text. Note that the baseline of a \texttt{\parbox} is set to the midpoint of the box, so if you include a \texttt{\parbox} in mid-sentence, the centre of the box will be lined up with the line of type currently being set. You can specify that the
top or bottom should align differently with respect to any surrounding text instead by adding an optional \textit{t} (top) or \textit{b} (bottom) in square brackets before the width. For example, \texttt{\parbox[t]{3in}{...}} will produce a box with the baseline aligned with the top line of the text in the box.

Where the contents is more complex, use the \texttt{minipage} environment.

\begin{minipage}{3in}
Please make sure you send in your completed forms by January 1st next year, or the penalty clause 2(a) will apply:
\begin{itemize}[noitemsep]
\item Incomplete forms will be returned to you unprocessed.
\item Forms must be accompanied by the correct fee.
\item There is no appeal. The adjudicators' decision is final.
\end{itemize}
\end{minipage}

Notice that when setting very narrow measures with type that is too large, the words will not fit nicely and the spacing may become uneven or there may be too much hyphenation. Either use \texttt{\raggedright} or reduce the type size, or (in extreme cases) reword the text or break each line by hand. Fortunately, it is rare for \LaTeX to need this level of attention.

Within a \texttt{minipage} you can use virtually everything that occurs in normal text (eg lists, paragraphs, tabulations, etc) with the exception of floats like Tables and Figures. The \texttt{minipage} environment takes a
compulsory argument just like \parbox does, and it means the same: the width you want the text set to.

Note that in both \minipage{}s and \parbox{}es, the paragraph indentation (\parindent) is reset to zero. If you need to change it, do so inside the \minipage{} or \parbox{} using the \setlength\command (see section 2.7 on page 49).

Because a \minipage{} is typeset independently from the rest of your text, any footnotes inside a \minipage{} will be typeset at the end of the \minipage{}, not at the foot of the containing page. They will also be done using lowercase letters by default, to keep them separate from the normal footnotes. (We haven’t done footnotes yet, but they’re in the next chapter.

There are other ways of typesetting text to widths other than the normal text width: you can use a one-row, one-cell \tabular{} environment with the p column type specification; or you can use the technique of setting the material into a special box that remembers it, and then emitting it where you want it (this is implemented by the standard \lrbox{} environment or by the Sbox environment from the fancybox package, but these are advanced techniques).

\subsection{Framed boxes}

To put a frame round \texttt{some text}, use the \fbox{} command:

\fbox{some text}

We already saw this used in the Quick Start document and also to frame an image in Figure 4.2 on page 90. For text, this works for a few words in mid-line, but the framed box and its contents won’t break over the end of a line. To typeset multiline text in a box, put it in a \parbox, or use a \minipage{} or \tabular{} environment as described above, and enclose the whole thing in a \fbox.

\fbox{\parbox{3in}{Please make sure you send in your completed forms by January 1st next year, or the penalty clause 2(a) will apply.}}

The spacing between text and box is controlled by the value of \fboxsep, and the thickness of the line by \fboxrule, both of which can be reset with the \setlength\command.
4.7. VERBATIM TEXT

If you are using colour, the xcolor package extends boxing to the \colorbox command, which takes two arguments: a colour name or code for the background colour, and the text (which will need a foreground colour if black would not be suitable):

\colorbox{green}{\textcolor{white}{some text}}

The package also provides \fcolorbox which puts a frame around a coloured box; in this case the first argument is the frame colour, the second the background colour, and the third the contents.

4.6.3 Sidebars and panels

The fancybox package lets you extend the principle of \fbox with commands to surround text in square, oval (round-cornered), and drop-shadow boxes (eg \ovalbox, \shadowbox, etc: see the documentation for details).

You can create panels of any size with these borders by using the minipage environment to typeset the text inside a special Sbox environment which fancybox defines. The minipage formats the text but the Sbox ‘captures’ it, allowing you to delay putting the frame around until it is complete.

The printed version of this document uses this extensively and there is a worked example shown in section 7.5 on page 168.

4.7 Verbatim text

If you are documenting computer procedures, you probably need fixed-width type for examples of programming or data input or output. Even if you are writing about completely non-computer topics, you may often want to quote a URI, filename, or email address which needs to be typeset specially.

\LaTeX includes two features for handling fixed-format text: inline verbatim and display verbatim. There are many more variations available in other packages.

4.7.1 Inline verbatim

To specify a word or phrase as verbatim text in typewriter type within a sentence, use the special command \verb, followed by the word or
phrase surrounded by any suitable character which does not occur in
the word or phrase itself. This is a very rare exception to the rule that
arguments go in curly braces.

For example, you could use the plus sign to show a $\LaTeX$ command in
a manual like this one:

```latex
You can typeset a phrase verbatim, even if it includes $\LaTeX$ command characters, for example the command to insert an image: \verb+\includegraphics[width=3in]{myhouse}+
```

The plus sign is ‘safe’ to use here because it doesn’t appear in
the code you want to typeset but you could use the grave accent or
backtick key ` or the vertical bar | if the phrase already had a
plus sign in it:

```latex
for example \verb\\(a=b+c)\ when illustrating the
$\LaTeX$ equation \(a=b+c\).
```

The \verb command has the advantage that it turns off all special
characters (see section 1.6 on page 15) except the one you use as
the delimiter, so you can easily quote sequences of characters in any
computer syntax — including $\LaTeX$. However, $\LaTeX$ will never break the
argument of \verb at a line-end when formatting a paragraph, even
if it contains spaces, so if it happens to be long, and falls towards
the end of a line, it will stick out into the margin. See section 1.9.2
on page 26 for more information on line-ends and hyphenation. The
argument to \verb MUST NOT contain a linebreak in your editor: this
will cause it to fail. See also the warning about using \verb inside
\footnote.

### 4.7.1.1 Typesetting URIs

The url package avoids this by performing a hyphenless break at
punctuation characters. It is particularly important in URIs to avoid
adding a spurious hyphen if they have to break over a line-end, because a hyphen might be mistaken by the user as a part of the address.4

URIs present another problem: it’s important for them to be visibly accurate, so they can be copied and retyped from print. It is therefore essential to use a typeface which distinguishes well between 1 (digit one), l (lowercase ell) and I (uppercase eye), and between 0 (zero) and O (uppercase oh). Monospaced ‘typewriter’ type usually makes this clear, but many sans-serif fonts do not. It is a common error by designers not to distinguish URIs in this way.

The url package provides the command \url which works in the same way as \verb, but uses the standard curly braces to enclose the address, eg \url{http://latex.silmaril.ie} — the command understands the syntax of a URI (Berners-Lee, Fielding & Masinter, 2005) and will never break mid-way through an unpunctuated word, only at slashes and full points (and never at embedded hyphens unless the hyphen package option is used). Bear in mind, however, that spaces and non-ASCII characters are (currently) forbidden in URIs, so using spaces in a \url argument will cause it to fail, as will using other non-URI-valid characters like accented letters.

4.7.1.2 Enhanced inline verbatim

The listings package, which we look at more below for display verbatim, also has an inline form. This can use colour to highlight your examples based on the language you are documenting — I am using it extensively in the PDF of this book.

The command \lstinline uses the same syntax as \verb (two matching but otherwise unused characters) to enclose the argument, but it provides for very extensive options to specify the language, font, size, style, and formatting. The most useful is the language, of which about 100 are predefined, from ADA to Verilog, and you can add new keywords and even whole new languages.

This is probably the most effective way to show computer-language examples inline, because it handles the syntax-based enhancement for you. It is, however, still subject to the same limitations as \verb, in

---

4 The original term Uniform Resource Locator (URL) is now deprecated in favour of the more accurate Uniform Resource Indicator (URI). For details see www.w3.org/Addressing/. Unfortunately the older term still persists, especially in this \LaTeX package and its command, and in some XML markup vocabularies.
that the code must fit on the space available in the line, or it will stick out into the margin.

For example, you could use the plus sign to show a \LaTeX command:
\begin{lstlisting}[language={\LaTeX}]
\verb+\includegraphics[width=3in]{myhouse}+
\end{lstlisting}
in order to display \includegraphics[width=3in]{myhouse}, because the plus sign does not occur in the command, and is therefore free to be used.

For longer (multiline) chunks of fixed-format text like examples of programming, use the \texttt{verbatim} environment. Like \verb, this turns off all special characters, so you can include anything at all in the verbatim text except the exact line \texttt{\end{verbatim}}, which MUST occur on a line by itself.

\begin{verbatim}
\documentclass[11pt,a4paper,oneside]{report}
\begin{document}
\title{Practical Typesetting}
\author{Peter Flynn\ Silmaril Consultants}
\date{December 2004}
\maketitle
\end{document}
\end{verbatim}

For more control over formatting there are two useful packages: the \texttt{verbatim} package, which overcomes a few of the limitations of the built-in \texttt{verbatim} environment; and the \texttt{fancyvrb} package, which provides much greater flexibility with a \texttt{Verbatim} environment (note the capital letter).

However, as I mentioned above, for a much more powerful verbatim environment, I use the \texttt{listings} package for its ability to colour the
keywords of a program according to the language used. It can also add rules, interpret internal formatting, and include external files, and let you add your own language definitions for new languages. The penalty is a slightly more complex configuration, but if you are documenting any kind of computer code in significant quantities, the quality and usability of the result is well worth it.

**Exercise 15. Try some fixed-format text**

1. Add your email address and home page URL using the `\verb` and `\url` commands. You’ll need to `\usepackage{url}` for the latter.

2. Load the `listings` package and try the `\lstinline` command to do the same.
Every text-handling system needs to support a repertoire of tools for doing things with text. \TeX{} implements many dozens, of which a small selection of the most frequently used is given here:

- footnotes and end-notes;
- marginal notes;
- cross-references, both normal ones and bibliographic citations;
- indexes and glossaries;
- typesetting in multiple columns.

### 5.1 Footnotes and end-notes

The command `\footnote{Like this}`, followed by the text of the footnote in curly braces, will produce an auto-numbered footnote with a raised small number where you put the command, and the numbered text automatically printed at the foot of the page.\footnote{Like this.} The number is reset to 1 at the start of each chapter (but there are packages to override that and make them run continuously throughout the document, or even restart at 1 on each page or section).
\LaTeX automatically creates room for the footnote, and automatically reformats it if you change your document in such a way that the point of attachment and the footnote would move to the next (or preceding) page.

\section*{Verbatim inside footnotes}

Because \LaTeX reads the whole footnote before doing anything with it, you can't use the $\verb$ (inline verbatim) command inside footnotes on its own: either use the $\verb!\VerbatimFootnotes!$ command from the \texttt{fancyvrb} package, or precede $\verb\footnote$ with $\verb!\protect$, or use (abuse?) the $\verb\url$ command instead (which you should be using for Web and email addresses in any case).

Footnotes in titling commands ($\verb!\title, \author$, etc) are generally regarded as Bad Style, and you should avoid them; if you can't, they produce the symbols $\star$, $\dagger$, $\ddagger$, $\S$, $\&$, $\|$, $\ast$, $\ddagger$, and $\ddagger\ddagger$ for the values 1–9 (and an error message for the tenth such footnote). In accordance with standard publishing practice, footnotes inside a \texttt{minipage} environment produce lettered notes instead of numbered ones, and they get printed at the bottom of the minipage, \emph{not} the bottom of the physical page (but this too can be changed).

There is a package (\texttt{endnote}) to hold over your footnotes and make them print at the end of the chapter instead or at the end of the whole document, and there is a package (\texttt{fnpara}) to print many short footnotes in a single footnoted paragraph so they take up less space. It is also possible to have several separate series of footnotes active simultaneously, which is useful in critical editions or commentaries in the Humanities: for example, a numbered series for the original author's original footnotes; a lettered series for footnotes by subsequent commentators or authorities in later editions; and a roman-numeral series for your own footnotes. It is also possible to format footnotes within footnotes.

If your footnotes are few and far between, you may want to use the sequence of footnote symbols above instead of numbers. You can do this by redefining the output of the footnote counter to be the $\verb!\fnsymbol!$ command (with the $\verb\footnote$ as its argument):
5.2 MARGINAL NOTES

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

There are also ways to refer more than once to the same footnote, and to defer the positioning of the footnote if it occurs in a float like a Table or Figure, where it might otherwise need to move to a different page, but these techniques are out of scope here.

5.2 Marginal notes

You can add marginal notes to your text instead of footnotes. You need or as well as to make sure that you have a wide-enough margin, of course: use the geometry package (see section 3.1.2 on page 57) to allocate enough space, otherwise the notes will be too cramped.

There are several packages to help with formatting marginal notes, but you can also define one yourself. Add this new command to your Preamble:

\newcommand{\marginal}[1]{% 
  \leavevmode\marginpar{\tiny\raggedright#1\par}}

Then you can use \marginal{Some text} where you need it. Be careful, however, because marginal notes are aligned with the line where the command starts, so a very long one followed too closely by another will cause \LaTeX{} to try and adjust the position so they don’t overlap.

We’re jumping ahead a bit here, as we haven’t covered how to define your own commands yet. I won’t even try to explain it here, although the astute reader can probably deduce it by inspection. See Chapter 7 starting on page 161 for more information about making up your own commands.

5.3 References and citations

This is one of the most powerful features of \LaTeX{}. As we mentioned when discussing Figures and Tables, you can label any point in a document with a name you make up, so that you can refer to it by that name from anywhere else in the document (or even from
another document) and \TeX will always work out the right cross-reference number for you, no matter how much you edit the text or move it around.

As we will see later, a similar method is also used to cite documents for a bibliography or list of references, and there are packages to sort and format these in the correct style for different journals or publishers.\(^2\)

### 5.3.1 Cross-references

You label the place in your document you want to refer to by adding the command \texttt{\label} followed by a short name you make up, in curly braces:\(^3\) exactly as we did for labelling Figures and Tables in section 4.2.2 on page 81.

```latex
\section{New Research}
\label{newstuff}
```

You can then refer to this place from anywhere in the same document with the command \texttt{\ref} followed by the name you used, eg

```latex
In section \ref{newstuff} there is a list of recent projects.
```

- Note the use of the unbreakable space (\texttt{~}) between the \texttt{\ref} and the word before it. This prints a space but prevents the line ever breaking at that point, should it fall close to the end of a line when being typeset.

- The \texttt{\S} command can be used if you want the section sign \texttt{\S} instead of the word ‘section’ (there is also a \texttt{\P} command that produces the paragraph sign or pilcrow \texttt{¶}).

Labels \textbf{MUST} be unique (that is, each value \textbf{MUST} occur only \textit{once} as a label within a single document), but you can have as many references

\(^2\) Be aware that in some disciplines where cross-references are not much used, the word ‘references’ may be used to mean ‘bibliographic references’.

\(^3\) This section is labelled \texttt{normalxref}, for example.
to them as you like. If you are familiar with HTML, this is the same concept as the internal linking mechanism using \# labels (or IDs in XHTML or HTML5).

**Labels in normal text:** If the label is in normal text, as above, the reference will give the current chapter/section/subsection number (depending on the current document class).

**Labels in Tables or Figures:** If the label is inside a Table or Figure, the reference provides the Table number or Figure number prefixed by the chapter number (remember that in Tables and Figures the \label command **MUST** come after the \caption command).

The \ref command does not produce the word ‘Figure’ or ‘Table’ for you: you have to type it yourself, or use the varioref package which automates it.

**Labels in lists:** A label in an item in an enumerated list will provide the item number. In other lists its value is null or undefined.

**Labels elsewhere:** If there is no apparent countable structure at the point in the document where you put the label (in a bulleted list, for example), the reference will be null or undefined.

The command \pageref followed by any of your label values will provide the page number where the label occurred, instead of the reference number, regardless of the document structure. This makes it possible to refer to something by page number as well as by its \ref number, which is useful in very long documents like this one (varioref automates this too).

Unresolved references are printed as two question marks, and they also cause a warning message at the end of the log file. There’s never any harm in having \label s you don’t refer to, but using \ref when you don’t have a matching \label is an error, as is defining two labels with the same value.

### 5.3.2 Bibliographic references

The mechanism used for references to reading lists and bibliographies is very similar to that used for normal cross-references. Instead

---

4 Thus I can refer here to the label at the start of this section as \ref {normalxref} and get the value ‘Section 5.3.1 on page 110’.
\LaTeX records the label values each time the document is processed, so the updated values will get used the next time the document is processed. You therefore need to process the document one extra time before final printing or viewing, if you have changed or added references, to make sure the values are correctly resolved. Most \LaTeX editors handle this automatically by typesetting the document twice when needed.

of using \texttt{\ref} you use \texttt{\cite} or one of the variants explained in section 5.3.2.3; and instead of \texttt{\label}, you attach a label value to each of the reference entries for the books, articles, reports, etc that you want to cite. You keep these reference entries in a bibliographic reference database that uses the \BibTeX data format (see section 5.3.2.2 on page 114).

This does away with the time needed to maintain and format references each time you cite them, and dramatically improves accuracy. It means you only ever have to enter the bibliographic details of your references once, and you can then cite them in any document you write, and the ones you cite will get formatted automatically to the style you specify (eg Harvard, Oxford, IEEE, Vancouver, MLA, APA, etc).

\subsection{Choosing between \LaTeX and \biblatex}

\LaTeX has two systems for doing citations and references, \LaTeX (old) and \biblatex (new): both of them use the same file format, also called \LaTeX. Both support the numeric and abbreviated alphabetic style formats built into \LaTeX plus a very wide range of others.

\LaTeX: the older \LaTeX has been in use for several decades and is still specified in many older document classes, especially journal article and book publishing formats. While it will continue to work, it has several drawbacks:

1. it doesn’t handle non-ASCII characters easily;
2. nor does the sort-and-extract program it uses (also called \texttt{bibtex});
5.3. REFERENCES AND CITATIONS

Bibliographic reference databases

Although it is possible to type the details of each reference manually, it’s much easier to use a program designed for the purpose. There are several available (see Wikipedia’s list), including Zotero and Mendeley. Both are open-source, but Mendeley was recently bought out by Elsevier; although it was still free of charge at the time of writing there have been concerns about Elsevier’s use of your data. Their features vary but Zotero’s primary benefit is that it can grab bibliographic metadata from web pages, so that you don’t have to type it in. Both can extract the metadata from the PDFs of articles you download from journal sites. And both can export the data in BibTeX format, which is essential.

Once your data is in BibTeX format, you can manage your collection of references with any of the many free BibTeX-based database programs such as JabRef (see Figure 5.1 on the next page).

Endnote and Reference Manager are commercial products which do not use the BibTeX data format, but which can export RIS format which Zotero, JabRef, and Mendeley can all import.

You add your entries to whichever system you choose, usually by downloading references from an online database like Web of Science, JSTOR, PubMed etc, or by using Zotero to gather the entry from a web page (you can also type references in by hand). JabRef lets you click an icon or menu entry and the LaTeX citation command will be inserted into your document editor at the cursor location.

3. the style format files (\texttt{.bst} files) are written in its own rather strange and unique language, making it extremely difficult to modify them or write new ones;

4. many of the style format files are old and out of date;

5. the range of data fields in references is limited and also out of date.

\texttt{biblatex} the newer \texttt{biblatex} system is now a well-established \LaTeX package to replace almost all of BibTeX. The main advantages are:
1. it works with UTF-8 characters, so non-ASCII, non-Latin, and other writing systems are handled natively, especially when using XeLaTeX.

2. there is a new sort-and-extract program (\texttt{biber}) to replace \texttt{bibtex}, which also handles UTF-8 natively.

3. the style format files are written entirely in LaTeX syntax, and under active development, so updating or writing layout formats it is much easier than with \texttt{BIBTEX}.

4. it extends the number and type of data fields that you can use in references.

5. it supports the popular author–year citation format natively, without the need for additional packages.

The only current drawback is that not all of the citation and reference formats supported in \texttt{BIBTEX} are yet available in \texttt{biblatex}, but the range is still wide, and modifications are usually easy to make.

The \texttt{biblatex} package with the \texttt{biber} program is therefore recommended, especially with XeLaTeX. However, as mentioned, there are a few classes and packages which have not yet been rewritten and may still require \texttt{BIBTEX} style formats, and there are some less common style formats which are not available for \texttt{biblatex}. However, the flexibility of the \texttt{biblatex} data model means altering or extending style formats to create your own is much easier than it is for \texttt{BIBTEX}, although this is not a task for the beginner.

\textbf{Cheatsheet}

Clea F Rees has written an excellent cheatsheet with virtually everything on it that you need for quick reference to using \texttt{biblatex}. This is downloadable as the package \texttt{biblatex-cheatsheet} from CTAN.

\subsection{The \texttt{BIBTEX} file format}

The format for \texttt{BIBTEX} files is used for both \texttt{BIBTEX} and \texttt{biblatex}, using either \texttt{biber} or \texttt{bibtex}. The file format is specified in the \texttt{original BIBTEX documentation} (look on your system for the file \texttt{btxdoc.pdf}). The
5.3. REFERENCES AND CITATIONS

The `biblatex` package and its updated style formats provide many more fields and document types.

Each Bib\TeX entry starts with an `@` sign followed by the name of the type of document, followed by the whole entry in a single set of curly braces. The first value is the unique key (label) that you make up, followed by a comma:

```latex
@book{fg,
  ...
}
```

Then comes each field (in any order), using the format:

```latex
fieldname = {value},
```

There MUST be a comma after each line of an entry except the last line:

```latex
@book{fg,
  title = {{An Innkeeper's Diary}},
  author = {John Fothergill},
  edition = {3rd},
  publisher = {Penguin},
  year = 1929,
  address = {London}
}
```

Some \TeX-sensitive editors have a Bib\TeX mode which understands these entries and provides menus or templates for writing them. The rules are:

- There MUST be a comma after each line of an entry except the last line;
- There MUST NOT be a comma after the last field in the entry (only — eg after `{London}` in the example);
- Some styles recapitalise the title when they format: to prevent this, enclose the title in double curly braces as in the example;
- Also use extra curly braces to enclose multi-word surnames, otherwise only the last will be used in the sort, and the others will be assumed to be forenames, for example the British explorer can be sorted under T with `author = {Ranulph {Twisleton Wykeham Fiennes}};`
CHAPTER 5. TEXTUAL TOOLS

- Multiple authors go in the one author field, separated by the word and (see example below);
- Values which are purely numeric (eg years) may omit the curly braces;
- Fields can occur in any order but the format must otherwise be strictly observed;
- Fields which are not used do not have to be included (so if your editor automatically inserts them as blank or prefixed by OPT [optional], you can safely delete them as unused lines).

There is a required minimum set of fields for each of a dozen or so types of document: book, article (in a journal), article (in a collection), chapter (in a book), thesis, report, conference paper (in a Proceedings), etc, exactly as with all other reference management systems. These are all (entry types and entry fields) listed in detail in the biblatex documentation (Lehman, Kime, Boruvka & Wright, 2015, sections. 2.1 & 2.2, 8).

Here’s another example, this time for a book on how to write mathematics — note the multiple authors separated by and.

```bibtex
@book{mathwrite,  
  author = {Donald E Knuth and Tracey Larrabee and Paul M Roberts},  
  title = {{Mathematical Writing}},  
  publisher = {Mathematical Association of America},  
  address = {Washington, DC},  
  series = {MAA Notes 14},  
  isbn = {0-88385-063-X},  
  year = {1989}}
```

Every reference in your reference database MUST have a unique key value (label or ID): you can make this up, just like you do with normal cross-references, but some bibliographic software automatically assigns a value, usually based on an abbreviation of the author and year. These keys are for your convenience in referencing: in normal circumstances your readers will not see them. You can see these labels in the right-hand-most column and at the bottom of the screenshot in Figure 5.1, and in the examples above. You use this label in your documents when you cite your references (see section 5.3.2.3 on page 117).

There are many built-in options to the biblatex package for adjusting the citation and reference formats, only a few of which are covered
here. Read the package documentation for details: it is possible to construct your own style simply by adjusting the settings, with no programming required (unlike the older Bib\TeX styles, which are written in a programming language used nowhere else).

Many users keep their Bib\TeX files in the same directory as their document[s], but it is also possible to tell \LaTeX and Bib\TeX that they are in a different directory. This is a directory specified by the \$\texttt{BIBINPUTS}$ shell or environment variable. On Unix & GNU/Linux systems (including Apple Macintosh OS X), and in \TeX Live for Windows, this is your \TeX installation’s \texttt{texmf/bibtex/bib} directory — the same one that old-style Bib\TeX .bst style files are kept in — but you should use your personal \TeX directory and create a subdirectory of the same name in there for your own .bib files. MiK\TeX also uses the same \$\texttt{BIBINPUTS}$ variable, but it is not set on installation: you need to set it using the Windows Systems Settings (see for example www.computerhope.com/issues/ch000549.htm).

### 5.3.2.3 Citation commands

The basic command is \texttt{\cite}, followed by the label of the entry in curly braces. You can cite several entries in one command: separate the labels with commas.

\begin{verbatim}
cite{fg}
cite{bull,davy,heller}
\end{verbatim}

For documents with many citations, use the \texttt{Cite} button or menu item in your bibliographic reference manager, which will insert the relevant command for you (you can see it activated for the \TeXStudio editor in Figure 5.1 on the preceding page).

How the citation appears is governed by two things:

1. the reference format (style) you specify in the options to the \texttt{biblatex} package (see section 5.3.2.4 on page 120) or in a \texttt{\bibliographystyle} command if you are using Bib\TeX (see section 5.3.2.5 on page 120) instead of \texttt{biblatex};

2. the type of citation command you use: \texttt{\cite, \textcite, \parencite, \autocite, \footcite}, etc, as shown below.

There are three built-in formats in \texttt{biblatex}:
There are two basic types of author–year citation:

**year in parentheses**: used in phrases or sentences where the name of the author is part of the sentence, and the year is there to identify what is being cited; in **biblatex** this command is `\textcite{fg}`

\[\ldots\text{as has clearly been shown by Fothergill (1929)}\].

This is sometimes called ‘author-as-noun’ citation.

**whole citation in parentheses**: used where the phrase or sentence is already complete, and the citation is being added in support: in **biblatex** this command is `\parencite{fg}`

\[\ldots\text{as we have already clearly shown (Fothergill, 1929)}\].

Note that author–year format is not built into **BibTeX** but can be done with the use of the **natbib** package and others.

**numeric**: This format is popular in some scientific disciplines and `\cite` produces just a number in square brackets [42]. The references at the end of the document may be numbered in order of reference or sorted by author.

**alphabetic**: This format is also popular in some scientific disciplines and `\cite` produces a three- or four-letter abbreviation of the author’s name and two digits of the year, all in square brackets [Fot29]. The references at the end of the document are listed with the the abbreviated key value as their label. This format is also called ‘abbreviated’.

To direct your reader to a specific page or chapter, you can add a prefix and/or a suffix as optional arguments in square brackets before the label.

\[\ldots\text{as shown by } \textcite[p 12]{mathwrite}.\]

A prefix gets printed at the start of the citation and the suffix gets printed at the end, but all still within the parentheses, if any. As they are both optional arguments, and as suffixes are far more common than prefixes, when only one optional argument is given, it is assumed to be the suffix. The example here therefore produces:

\[\ldots\text{as shown by Knuth et al. (1989, p. 12)}.\]
Table 5.1: Built-in citation style commands and formats (\texttt{biblatex})

<table>
<thead>
<tr>
<th>Style</th>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>authoryear</td>
<td>\parencite{fg}</td>
<td>(Fothergill, 1929)</td>
</tr>
<tr>
<td>authoryear</td>
<td>\textcite{fg}</td>
<td>Fothergill (1929)</td>
</tr>
<tr>
<td>authoryear</td>
<td>\footcite{fg}</td>
<td>Fothergill 1929</td>
</tr>
<tr>
<td>numeric</td>
<td>\cite{fg}</td>
<td>[42]</td>
</tr>
<tr>
<td>alphabetic</td>
<td>\cite{fg}</td>
<td>[Fot29]</td>
</tr>
<tr>
<td>authoryear</td>
<td>\cite{fg}</td>
<td>Fothergill 1929</td>
</tr>
</tbody>
</table>

¹ Fothergill 1929.

If you are using \texttt{bibtex} instead of \texttt{biblatex}, you can only specify a suffix.

Footnoted citations are common in History and related disciplines, to the extent that scholars in these fields actually call their references ‘footnotes’, which is confusing to others. The command \texttt{\footcite} does these (see Table 5.1 on page 119) but it is only relevant for author-year styles (in numeric style it just produces the number, which would be misleading).

There are many variant forms of the citation commands, either for specific styles like Harvard, IEEE, APA, MLA, etc; or for grammatical modifications like capitalising name prefixes, omitting the comma between name and year, or adding multiple notes; or for extracting specific fields from an entry (eg \texttt{\titlecite}). If you have requirements not met by the formats described here, you can find them in the documentation for the \texttt{biblatex} package.

Modern Language Association (MLA) citation is a special case, as it omits the year and instead \texttt{requires} the location of the citation within the document (eg the chapter, section, page, or line). It may include the title, if there would otherwise be ambiguity. The \texttt{biblatex} format for MLA citation handles the context-dependent formatting with the command \texttt{\autocite}; for \texttt{\LaTeX} there is an old version of MLA implemented in the \texttt{mla} and \texttt{hum2} packages.

If you are using \texttt{\LaTeX} instead of \texttt{biblatex}, the commands you can use are not standardised except for \texttt{\cite}. Instead, they depend on which style format you use; for example the popular \texttt{natbib} package, which

---

\footnotesize

\textbf{Formatting Information}
implements author-year citation for the natural sciences, uses \cite and \citet instead of \textcite and \parencite.

**Figure 5.1:** JabRef displaying a file of references, ready to insert a citation of Fothergill’s book into a LaTeX document being edited with TeXStudio.

Your reference management software will have a display something like Figure 5.1 on page 119 (details vary between systems, but they all do roughly the same job in roughly the same way), showing all your references with the data in the usual fields (title, author, date, etc).

Your BibTeX file, which contains all your bibliographic data, can be saved or exported as a .bib file from most reference management software (JabRef uses this format natively), It looks like the examples in section 5.3.2.2 on page 114. Your .bib file works with both biblatex and, but biblatex provides more field types and document types so that your references can be formatted more accurately.

If your bibliographic management software doesn’t save BibTeX format direct, save your data in RIS format, then import the .ris file into JabRef and save it as a .bib file from there.

### 5.3.2.4 Setting up biblatex with biber

You set up your document with the following packages:
5.3. REFERENCES AND CITATIONS

1. the \texttt{babel} package with appropriate languages, \textit{even if you are only using one language}. The default language is American English, so there are commands to map this to other language variants (the example below shows this for British English);

2. the \texttt{csquotes} package, which automates the use of quotation marks around titles or not, depending on the type of reference;

3. the \texttt{biblatex} package itself, specifying the \texttt{biber} program and the style of references you want, either \texttt{numeric}, \texttt{alphabetic}, or \texttt{authoryear}; or a publisher's style (in this example I am using \texttt{APA} format); and any options for handling links like DOIs, URIs, and ISBNs;

4. the language mapping command, if needed (see the documentation for the style you have chosen to find out if you need this);

5. finally, the name of your \LaTeX{} file(s) (see the panel 'Bibliographic reference databases' on p. 113) with one or more \texttt{\addbibresource} commands.

\begin{verbatim}
\usepackage[frenchb,german,british]{babel}
\usepackage{csquotes}
\usepackage[backend=biber,doi=true,isbn=true,url=true,style=apa]{biblatex}
\DeclareLanguageMapping{british}{british-apa}
\addbibresource{myrefs.bib}
\end{verbatim}

At the end of your document you need to add the \texttt{\printbibliography} command (or elsewhere at the point in your document where you want the full list of references you have cited to be printed). See section 5.3.2.6 on page 122 for details of how \LaTeX{} produces the references.

5.3.2.5 Using \BibTeX{}

The principles underlying \BibTeX{} are identical to \texttt{biblatex}: you create and maintain your \BibTeX{} file of references in exactly the same way, and you use the \texttt{\cite} in the same way. But there is no basic package to load with options: instead, you specify the name of the style you want,
CHAPTER 5. TEXTUAL TOOLS

**Versions of biblatex and biber**

One critically important point to note is that biblatex and biber are step-versioned; that is, each version of the biblatex package only works with a specific version of the biber program. There is a table of these dependencies in the biblatex documentation PDF. If you manually update biblatex for some reason (perhaps to make use of a new feature), you MUST also update your copy of biber to the correct version, and vice versa; otherwise you will not be able to produce a bibliography.

The BibTeX method is nowadays deprecated, partly because it does not handle UTF-8 character-encoding correctly, and partly because its unusual internal programming language makes it hard to extend.

Using a `\bibliographystyle` command in your Preamble, plus any additional packages needed to help with the formatting. You then use the `\bibliography` command to give the name of your BibTeX file (without the `.bib` extension) at the point you want the references to be printed.

\begin{verbatim}
\bibliographystyle{apsr}
\usepackage{natbib,har2nat}
...
\bibliography{myrefs}
\end{verbatim}

In this example the American Political Science Review (APSR) variant of the Harvard reference style has been selected (common in the political and economic sciences), which normally means numerical citation, but the author wants the Natural Sciences (author-year) form of citation, which is provided by the natbib package, which in turn requires the har2nat package to handle Harvard-style formatting. As explained above, the natbib package provides \citet for textual citation like Fothergill (1929), and \citep for parenthetic citation like (Fothergill, 1929).
There is an option on all the \texttt{cite} commands and variants to let you specify a suffix to the citation, so \texttt{\citep{Foreword, p.13}}} produces (Fothergill, 1929, p. 13, Foreword).

See section 5.3.2.6 on page 122 for details of how PDF\LaTeX{} produces the references.

5.3.2.6 Producing the references

Because of the record\texttt{extract}\texttt{format} process (the same as used for cross-references), you will get a warning message about ‘unresolved references’ the first time you process your document after adding a new citation for a previously uncited work and running \texttt{biber} or \texttt{biblatex}. The \texttt{biblatex} program produces a bold ?? where the unresolved reference will be; \texttt{biber} produces the entry label in bold instead. This will disappear once \LaTeX{} has been run again, which is why most editors have a \texttt{Build} function to do the job for you.

Your \LaTeX{} editor’s \texttt{Typeset} or \texttt{Build} button or menu entry should therefore handle the business of running \texttt{biber} or \texttt{biblatex} for you. If not, here’s how to do it manually in a Command window:

For \LaTeX{} with \texttt{biber}: Run \LaTeX{}, then run \texttt{biber} to extract and sort the details from the \texttt{BIB} file, and then run \LaTeX{} again:

\begin{verbatim}
xelatex myreport
biber myreport
xelatex myreport
\end{verbatim}

For PDF\LaTeX{} with \texttt{biblatex}: Run PDF\LaTeX{}, then run \texttt{biblatex} to extract and sort the details from the \texttt{BIB} file, and then run PDF\LaTeX{} again twice (to resolve the references):

\begin{verbatim}
pdflatex myreport
biblatex myreport
pdflatex myreport
pdflatex myreport
\end{verbatim}

In practice, authors tend to retypeset their documents from time to time during writing anyway, so they can keep an eye on the typographic progress of the document. Just clicking the \texttt{Build} button after adding a new \texttt{\cite} command, and subsequent runs of \LaTeX{}
CHAPTER 5. TEXTUAL TOOLS

will incrementally incorporate all references without you having to worry about it.

If you work from the command line, the \texttt{latexmk} script automates this, running \texttt{bibtex} or \texttt{biber} and re-running \LaTeX{} again when needed.

5.4 Indexes and glossaries

Indexes and glossaries are tools for directing or helping the reader. Any book or report sized document should have an index, although they are uncommon in theses. Glossaries are usually only needed where there is a substantial number of technical terms needing formal definition and cross-referencing.

5.4.1 Indexes

\LaTeX{} has an automated indexing facility which uses the standard \texttt{makeindex} program for sorting and collation. To use indexing, use the package \texttt{makeidx} and include the \texttt{\makeindex} command in your Preamble to initialise the index:

\begin{quote}
\begin{verbatim}
\usepackage{makeidx}
\makeindex
\end{verbatim}
\end{quote}

When you want to index something, use the command \texttt{\index{entry}} followed by the entry in curly braces, as you want it to appear in the index, in one of the following formats:

Plain entry: Typing \texttt{\index{beer}} will create an entry for ‘beer’ with the current page number;

Subindex entry: For an entry with a subentry use an exclamation mark to separate them: \texttt{\index{beer!lite}}. You can create another level as well, so you can have subsubentries like \texttt{\index{beer!lite!American}};

Cross-references: ‘See’ entries are done with the vertical bar (one of the rare times it does not get interpreted as a math character): \texttt{\index{Microbrew!see{beer}}};

Font changes: To change the style of an entry, use the \@-sign followed by a font change command:
This example indexes *Chocolate Stout* as a third-level entry and italicises it at the same time. Any of the standard \text font-change commands work here: see Table 6.7 on page 150 for details.

You can also change the font of the index number on its own, as for first-usage references, by using the vertical bar in a similar way to the ‘see’ entries above, but substituting a font-change command name alone (without a backslash or curly braces) such as \textbf for bold-face text (see the index):

\index{beer!Rogue!Chocolate Stout|textbf}

**Out of sequence**: The same method can be used as for font changes, but using the alternate index word instead of the font command name, so \index{Oregon Brewing Company@Rogue} will add an entry for ‘Rogue’ in the ‘O’ section of the index, as if it was spelled ‘Oregon Brewing Company’.

When the document has been processed through \LaTeX\ it will have created a .idx file, which you run through the \texttt{makeindex} program by clicking the \texttt{Index} button or menu entry in your editor, or by typing the \texttt{makeindex} command followed by your document name without the .tex filetype.

The program will look for the .idx file with the same name as your document, and output a .ind file with the sorted index in it. This is what gets used by the command \texttt{printindex} which you put at the end of your document, where you want the index printed. The default index format is two columns with a space between letters of the alphabet. The Unix manual page\textsuperscript{6} for the \texttt{makeindex} program has details of how to add letter headings to each alphabet group.

\textsuperscript{6} On Unix & GNU/Linux systems (including Apple Macintosh OS X, just type the command $\texttt{man makeindex}$; the page is also available in many reference sites on the web. 

\textit{Formatting Information}
5.4.2 Glossaries

Glossaries can be done in a similar manner to indexes, using the command `\makeglossary` in the Preamble and the command `\glossary` in the same way as `\index`. There are some subtle differences in the way glossaries are handled: both the books by Lamport (1994) and by Mittelbach, Goossens, Braams, Carlisle and Rowley (2004) duck the issue, but there is some documentation on `glotex` on CTAN. There is also a `gloss` package based on BiBTeX which uses `\gloss` in the same way as `\cite`.

However, by far the best way is to use the `glossaries` package (not `glossary`, which is obsolete, and not `gloss`). This is a relatively complex package, as glossaries are a relatively complex tool, but there is extensive help in the documentation. It requires the `makeglossaries` script from CTAN (there is also a `makeglossariesgui` Java GUI).

Basically, you need to create a set of definitions, one per item to be glossed, using the `\newglossaryentry` command:

\begin{verbatim}
\newglossaryentry{esis}{name={ESIS},description={The \textbf{Element Structure Information Set} of a marked-up document, originally defined for SGML (replaced for XML using W3C Schemas by the Post-Schema-Validation InfoSet, PSVI). See \url{http://xml.coverpages.org/WG8-n931a.html}}}
\end{verbatim}

This specifies a) the label you will use (`esis`); b) the name of the item as it will be printed (`ESIS`); and c) the textual description to go in the glossary. Probably the best place to put these is in a separate file like `mygloss.tex`, which you can get \LaTeX{} to read with an `\input{mygloss.tex}` command in your Preamble.

You can then use the `\gls` command in your text to produce the printable name of any entry, using the label to refer to it, e.g. `\gls{esis}`. It is possible to use or define variant commands to handle references at the start of a sentence (where you need a capital letter if the name is not an acronym), and grammatical alteration like unusual plurals or forms ending in ‘—ing’.

At the end of your document, where you want the glossary printed, you use the command `\printglossaries`. The glossary need to be processed separately from the main document, using the `makeglossaries` script
script, exactly the same way as you do for \texttt{biber}, \texttt{bibtex}, and \texttt{makeindex}. The Build function of your editor should do this for you, or use a Makefile or a utility like \texttt{latexmk}. As with all these tools, there are many more facilities built into them: read the documentation.

The \texttt{acronym} package can also be used to create a kind of glossary containing a list of acronyms and their expansions, as well as driving the use of expansion on first mention. However, the \texttt{glossaries} package now contains built-in support for acronyms.
5.5 Multiple columns

Use the `multicol` package: the environment is called `multicols` (note the plural form) and it takes the number of columns as a second argument in curly braces:

\usepackage{multicol}
...
\begin{multicols}{3}
...
\end{multicols}

\LaTeX{} has built-in support for two-column typesetting via the `twocolumn` option in the standard Document Class Declarations, but it is relatively inflexible in that you cannot change from full-width to double-column and back again on the same page, and the final page does not balance the column heights. However, it does feature special `figure*` and `table*` environments which type-set full-width figures and tables across a double-column setting.

The more extensive solution is the `multicol` package, which will set up to 10 columns, and allows the number of columns to be changed or reset to one in mid-page, so that full-width graphics can still be used. It also balances the height of the final page so that all columns are the same height — if possible: it’s not always achievable — and you can control the width of the gutter by setting the `\columnsep` length to a new dimension.

Multi-column work needs some skill in typographic layout, though: the narrowness of the columns makes typesetting less likely to fit smoothly because it’s hard to hyphenate and justify well when there is little space to manoeuvre in.
6

Layouts and fonts

This is the chapter that most users think they want first, because they come to structured documents from a wordprocessing environment where the only way to convey different types of information is to fiddle with the font and size drop-down menus.

As you will have seen by now, this is normally unnecessary in \LaTeX, which does most of the work for you automatically. However, there are occasions when you need to make manual typographic changes, and this chapter is about how to do them.

6.1 Changing layout

The design of the page can be a very subjective matter, and also a rather subtle one. Many organisations large and small pay considerable sums to designers to come up with page layouts to suit their purposes. Styles in page layouts change with the years, as do fashions in everything else, so what may have looked attractive in 1978 or 1991 may look rather dated in 2020.

As with most aspects of typography, making the document readable involves making it consistent, so the reader is not interrupted or distracted too much by apparently random changes in margins, widths, or placement of objects.\footnote{Some authors — and perhaps some designers — believe that consistency is undesirable, and that double-page layouts in printed books should each be designed inde-} However, there are a number of different...
occasions where the layout usually does change, related to the frequency with which the format appears.

- In books, the title page, the half-title, copyright and legal pages, dedication, acknowledgements, and other one-page preliminaries (if you use them) are usually designed individually, as the information on them only occurs once in that format anywhere in the document.

- The Table of Contents and related lists like the List of Figures and List of Tables all need to share one design.

- Longer prelms like a Foreword, Introduction, or Preface should likewise follow the same format between them.

- Chapter and Appendix start pages always share a layout.

- Other (normal) pages have a single layout, but within the page there will be individual variations to handle tables, lists, figures, sidebars, exercises, footnotes, etc.

The exceptions to this are newspapers and magazines, where page layout is done individually, page by page (or pairs of facing pages together), but even here, most publications have strict rules about what blocks of material can be placed where, and use a carefully-designed set of templates to achieve this.

If you are going to design a whole document yourself, it’s probably a good idea to read a couple of books on layout design first, to get a feel for the conventions which contribute to making the reader comfortable reading.

While unusual or radical layouts have an important role in attention-grabbing, or in making a socio-political statement (‘WIRED’ magazine is an obvious example), they are usually out of place in business reports, white papers, books, theses, and journals. In ephemera, on the other hand, as in advertising, they are probably critical.

Kirschenbaum’s magnificent Goodbye Gutenberg expresses this both eloquently and attractively, but the cost of such design labour and the cost of four-colour printing on all pages places it beyond the reach of most publishers’ budgets until the economics of on-demand four-colour ‘printing’ makes it possible.
6.1 Changing Layout

6.1.1 Margins and spacing

We mentioned in section 5.2 on page 109 and elsewhere the existence of the geometry package which lets you change margins. It also lets you set the text-area height and width and a lot of other layout settings: read the documentation for details (see section 3.1.3 on page 59 for how to read package documentation). Here is an example:

\usepackage[a4paper, left=2cm, top=1cm, bottom=2cm, right=3cm, nohead, nofoot]{geometry}

There is a section on using the geometry package that you only need to specify some of the margins or the text height/width; once it knows the paper size, if you give it the text width and the left-hand margin, for example, it can work out the right-hand margin. The package also provides the \textwidth command, which lets you over the margin settings in mid-document. This probably isn't something you want to do very often, though.

The spacing around the individual textual components (paragraphs, lines, footnotes, tables, figures, etc.) can also be changed on a document-wide basis, as we use with paragraph spacing and indentation in section 2.7 on page 47. There are a lot of packages available to do various aspects of this, too; you can go into the LaTeX Source Code Archive (LSAC) to find what you need.

Changing the spacing of section headings for the whole document can be done with the \renewcommand package, designed for you adjust section-head spacing without having to know about the internal \hsize scaling, which is quite complex. The spacing for lists can be adjusted with the \setlength package. In both cases the user with highly specific requirements such as a publisher's Compsetter's specification should read the relevant sections in the Companion or ask for expert help, as there are many more settings which can also be changed in the text on your design, but which need some understanding of \hsize internals.

There are almost infinite possibilities for changing changes so that they occur every time in a consistent manner. You can also make manual changes every time you write the same text:

\begin{itemize}
  \item Left: set to 8cm\times28cm
  \item Right: set to 17cm\times22cm
\end{itemize}

Bear in mind when using the geometry package that you only need to specify some of either the margins or the text height/width. Once it knows the paper size, if you give it the text width and the left-hand margin, for example, it can work out the right-hand margin.
The package also provides the \texttt{newgeometry} command, which lets you reset the margin settings in mid-document. This probably isn’t something you want to do very often, though.

The spacing around the individual textual components (paragraphs, lists, footnotes, tables, figures, etc) can also be changed on a document-wide basis, as we saw with paragraph spacing and indentation in section 2.7 on page 50. There are a lot of packages available to do various aspects of this, far too many to go into detail here: search CTAN to find what you need.

Changing the spacing of section headings for the whole document can be done with the sectsty or section packages, designed to let you adjust section-head spacing without having to know about the internal \LaTeX coding, which is quite complex.

The spacing for lists can be adjusted with the enumitem package. In both cases the user with highly specific requirements such as a publisher’s Compositor’s Specification should read the relevant sections in the Companion or ask for expert help, as there are many extra settings which can also be changed to fine-tune your design, but which need some understanding of \LaTeX’s internals.

All the above are for automating changes so that they occur every time in a consistent manner. You can also make manual changes:

**Flexible vertical space:** There are three commands \texttt{smallskip}, \texttt{medskip}, and \texttt{bigskip}. These output flexible (dynamic, or ‘rubber’) space, approximately 3pt, 6pt, and 12pt high respectively, so they will automatically compress or expand a little, depending on the demands of the rest of the page (for example to allow one extra line to fit, or a heading to be moved to the next page without anyone except a typographer noticing the change). These commands can only be used after a paragraph break (a blank line or the command \texttt{par}).

**Fixed vertical space:** For a fixed-height space which will not stretch or shrink, use the command \texttt{vspace} followed by a length in curly braces, eg \texttt{vspace{18pt}} (again, this has to be after a paragraph break). Bear in mind that extra space which ends up at a page-break when the document is formatted will get discarded entirely to make the bottom and top lines fall in the correct places. To force a vertical space to remain and be taken into account...
6.1. CHANGING LAYOUT

even after a page break (very rare), use the starred variant, eg \vspace*{19pt}.

Double line-spacing: \LaTeX’s \baselinestretch value governs the amount of extra line-spacing based on the current font size (see section 6.2.7 on page 151). By default it is null, meaning no extra space. It is possible to set it to a multiplier, like \renewcommand{\baselinestretch}{1.75} to make it 1.75 times normal. However...

Double-spacing normal lines of text is usually A Bad Idea, as it looks very ugly, but increased line-spacing does become important if you are typesetting very wide lines, otherwise the reader’s eye will not be able to pick up the start of a new line easily.

Double-spacing is still a requirement in many universities for thesis submission, partly because of the tendency of writers to use very wide lines on office-type paper sizes, and partly because the reviewers needed space to write in corrections. With the growth of electronic submission and editorial corrections in PDF files, it should become less necessary. Nowadays, 1\³ or 1\½ line spacing is considered acceptable, according to your font size.

Use the \setspace package to do this. It has commands \doublespacing for double line-spacing and \onehalfspacing for one-and-a-half line spacing (the command \singlespacing resets them). There is also a \spacing environment to let you specify a different multiple as the argument:

\begin{spacing}{1.333}
...
\end{spacing}

Be aware that you may not want footnotes to be spaced by the same multiple as your normal text, and you may want other elements like lists, tables, figures, or quotations spaced differently.

As with theses, there are some perfectly genuine and normal reasons for wanting bigger line spacing, for example when typesetting a proof of a critical or variorum edition, where editors and contributors are going to want to add notes manually, or...
where the text is going to be overprinted by something else like Braille, or in advertising or display text for special effects.

**Horizontal space**: There is a horizontal equivalent to the \texttt{vspace} command which works in the same way, so \texttt{hspace\{1in\}} will insert a 1” space like this in mid-paragraph. There are also some predefined (shorter) spaces available:

- \texttt{thinspace} (\(\frac{1}{6}\)em), which we saw between single and double quotes in section 1.7 on page 18. It’s also sometimes used between the full point after abbreviations and a following number, as in page references like p. 42, where a word space would look too big, and setting it solid would look too tight.

- \texttt{enspace} (\(\frac{1}{2}\)em). There is no direct equivalent predefined in \LaTeX for ‘mid’ and ‘thick’ spaces as used by metal typesetters, although it would be possible to define them. The en as a unit is used as the width of a single digit in many fonts, as a convenience so that numbers in listings are easier to line up.

- \texttt{quad} (1em) was originally the width of a capital M in metal type.

- \texttt{qquad} (2em) is double a \texttt{quad}.

Beyond this, all horizontal space within paragraphs is automatically flexible, as this is what \LaTeX uses to achieve justification. Never be tempted to try and change the spacing between letters unless you have some professional training in typography. Some systems use adjustable inter-letter spacing (incorrectly called ‘tracking’) as an aid to justification and it is almost always wrong to do so (and looks it). While it is of course possible to change letterspacing in \LaTeX (with the \texttt{soul} package), it should only be done by a typographer, and then only very rarely, as the settings are very subtle and beyond the scope of this book.\(^2\)

\(^2\) This does not apply for the German technique in blackletter type of using letter-spacing instead of (non-existent) italics. The defaults in the \texttt{soul} package were designed to cater for this.
6.1.2 Headers and footers

\LaTeX{} has built-in settings to control the page style of its default page layouts, and space at the top and bottom of the page is provided automatically for them (it can also be adjusted or turned off in the \texttt{geometry} package). These settings are implemented with the \texttt{\pagestyle} command, which can take one of the following arguments in curly braces:

- \texttt{plain} for a page number centered at the bottom — this is the default;
- \texttt{empty} for nothing at all, not even a page number — use this when you are doing one-page documents like posters or handouts, where a page number has no meaning;
- \texttt{headings} for running heads based on the current chapter and section — this is common for articles, books, and reports, so that every page is identifiable even if extracted or printed or copied separately;
- \texttt{myheadings} lets you use your own \texttt{\renewcommand}\texttt{\markright} and \texttt{\markboth} commands, which control how chapter and section titles get into page headers.

The command \texttt{\thispagestyle} (taking the same arguments) can be used to force a specific style for the current page only.

However, the easiest way to get specialist running heads is to use the \texttt{fancyhdr} package, which lets you redefine the left-hand, centre, and right-hand headers and footers for both odd-numbered (left-hand) and even-numbered (right-hand) pages (twelve objects in all).

These areas can contain a page number, fixed text, variable text (like the current chapter or section title, or the catch-words of a dictionary), or even a small image. They can also be used to do page backgrounds and frames, by making one of them the top corner of an invisible box which ‘hangs’ text or images down over the whole page.

The settings for the typeset version of this document can be used as an example: for the whole story you have to read the documentation.

\begin{verbatim}
\pagestyle{fancy}\fancyhead{}
\renewcommand\headrulewidth{.1pt}
\fancyhead[LO,RE]{\footnotesize\sffamily\leftmark}
\end{verbatim}
TABLE 6.1: HEADER AND FOOTER LOCATIONS IN THE FANCYHDR PACKAGE

<table>
<thead>
<tr>
<th>Top left, even</th>
<th>Top centre, even</th>
<th>Top right, odd, even</th>
<th>Top left, odd</th>
<th>Top centre, odd</th>
<th>Top right, odd, even</th>
</tr>
</thead>
<tbody>
<tr>
<td>LH page, even-numbered</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bottom left, even</th>
<th>Bottom centre, even</th>
<th>Bottom right, even</th>
<th>Bottom left, odd</th>
<th>Bottom centre, odd</th>
<th>Bottom right, odd</th>
</tr>
</thead>
<tbody>
<tr>
<td>RH page, odd-numbered</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\fancyhead[LE, RO]{\footnotesize sffamily \itshape rightmark}
\fancyfoot[C]{}
\fancyfoot[LE, RO]{\setlength{\fboxsep}{2pt}\ovalbox{\footnotesize sffamily \thepage}}
\fancyfoot[LO, RE]{\footnotesize sffamily @title}
\fancypagestyle{plain}\fancyhf{}
\fancyfoot[R]{\setlength{\fboxsep}{2pt}\ovalbox{\footnotesize sffamily \thepage}}
\fancyfoot[L]{\footnotesize sffamily @title}
\renewcommand{\headrulewidth}{0pt}

This is probably more complex than most documents, but it illustrates some common requirements:

1. Settings are prefixed by making the \texttt{\textbackslash pagestyle} `fancy’ and setting the \texttt{\textbackslash fancyhead} to null to zap any predefined values.

2. The thickness of the rule at the top of the page can be changed (or set to 0pt to make it disappear).

3. The header and footer settings are specified with L, C, and R for left, centre, and right; and with O and E for Odd and Even numbered pages. In each setting, the typeface style, size, and font can be specified along with commands which implement various
6.2. USING FONTS

The default typeface in \LaTeX{} is Computer Modern (CM). This typeface was created by Knuth for use with \TeX{} — it is based on a Victorian book typeface, Monotype Series 8, and he designed \TeX{} originally for typesetting books. Because it is one of the very few book typefaces with a comprehensive set of fonts, including a full suite of mathematics, it has remained the default, rather than the variations on Times you find in wordprocessors, because until recently the mathematical symbols for Times were a commercial product.

Computer Modern is based on a 19th-century book typeface from Monotype, which is why it looks a little like an old-fashioned school book. This paragraph is set in Computer Modern so you can see what it looks like. The typeface was designed using METAFONT, the font-drawing program made by Knuth to accompany \TeX{} systems, but it is now also available in Type 1 and TrueType formats.

\LaTeX{} also comes with the popular (some would say overused) ‘Adobe 35’ fonts (see Table 6.3 on page 143) which are built into PDF readers, laser printers, and other DTP systems, plus some extra fonts donated by the X Consortium and individuals. There are many other fonts designed using METAFONT which can be downloaded from CTAN, including a large collection of historical, symbol, initial, and non-Latin fonts (META-Font creates Type 3 bitmap fonts for PDFs, but many of them are now available in Type 1 outline format as well).

\LaTeX{} can use any of the Type 1 fonts available both free and commercially, and Xe\LaTeX{} can use any of the TrueType and OpenType fonts as well.

6.2.1 Setting up fonts for Xe\LaTeX{}

Having said that Xe\LaTeX{} lets you use all your system fonts (as well as the ones that come with \TeX{}), there is one small piece of preparation to do first: indexing them (properly speaking, cache them). Without doing
this, you can still use them but you would have to tell \TeX{} where to
find each one, which is tedious.

The following details are for GNU/Linux systems; details for Apple
Mac OS X and Microsoft Windows are still being researched.

1. Open a Command or Terminal window;
2. Become root by typing \texttt{\textasciitilde}$sudo \textasciitilde$ su – and giving your password
   when asked;
3. Run your favourite text editor (eg \texttt{emacs}, \texttt{vi}, \texttt{kate}, \texttt{gedit}, etc);
4. Create a new file in \texttt{/etc/fonts/conf.avail} called
   \texttt{09-texlive.conf};
5. If you installed \TeX{} from the distribution’s own package repositori-
   es, copy and paste this content into the file:

```xml
<?xml version="1.0"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<fontconfig>
  <dir>/usr/share/texlive/texmf-dist/fonts/opentype</dir>
  <dir>/usr/share/texlive/texmf-dist/fonts/truetype</dir>
  <dir>/usr/share/texlive/texmf-dist/fonts/type1</dir>
  <dir>/usr/share/texlive/texmf/fonts/opentype</dir>
  <dir>/usr/share/texlive/texmf/fonts/truetype</dir>
  <dir>/usr/share/texlive/fonts/type1</dir>
  <dir>/usr/share/fonts/opentype</dir>
  <dir>/usr/share/fonts/truetype</dir>
  <dir>/usr/share/fonts/type1</dir>
  <dir>/usr/share/doc</dir>
</fontconfig>
```

If you installed \TeX{} from the \TeX{} Live distribution from \texttt{CTAN}, copy
and paste this content instead:

```xml
<?xml version="1.0"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<fontconfig>
  <dir>/usr/local/texlive/2015/texmf-dist/fonts/opentype</dir>
  <dir>/usr/local/texlive/2015/texmf-dist/fonts/truetype</dir>
  <dir>/usr/local/texlive/2015/texmf-dist/fonts/type1</dir>
  <dir>/usr/local/texlive/texmf-local/fonts/opentype</dir>
  <dir>/usr/local/texlive/texmf-local/fonts/truetype</dir>
  <dir>/usr/local/texlive/texmf-local/fonts/type1</dir>
  <dir>/usr/share/fonts/opentype</dir>
  <dir>/usr/share/fonts/truetype</dir>
</fontconfig>
```
6.2 USING FONTS

6.2.1 Installing fonts with \fontspec

6.2.2 Changing the default font family with \fontspec

Formatting Information
\usepackage{fontspec}  
\setmainfont{GFS Bodoni}  
\setsansfont{Gill Sans}  
\setmonofont{Everson Mono}

Most of the time that’s all you need. Unfortunately, not all OT/TT fonts have well-formed names, so there are ways to help \TeX find them. The commands above should find the fonts provided they are installed in your TEX distribution directories or your personal TEX directory.

Both TT and OT fonts are usually named in a pattern where the base name is followed by the variant (regular, italic, bold, bold-italic, etc) separated by a hyphen or underscore character. For example, this is the directory listing of my installation of the Liberation typeface:

```
-rw-r-r-- 1 peter 152660 May 5 2014 LiberationSerif-BoldItalic.ttf
-rw-r-r-- 1 peter 147132 May 5 2014 LiberationSerif-Bold.ttf
-rw-r-r-- 1 peter 152408 May 5 2014 LiberationSerif-Regular.ttf
-rw-r-r-- 1 peter 139764 May 5 2014 LiberationSans-Italic.ttf
-rw-r-r-- 1 peter 133540 May 5 2014 LiberationSansNarrow-BoldItalic.ttf
-rw-r-r-- 1 peter 113304 May 5 2014 LiberationSansNarrow-Regular.ttf
-rw-r-r-- 1 peter 115964 May 5 2014 LiberationMono-Bold.ttf
-rw-r-r-- 1 peter 127420 May 5 2014 LiberationMono-Italic.ttf
-rw-r-r-- 1 peter 108492 May 5 2014 LiberationMono-Regular.ttf
```

In this example you can see four families of the Liberation typeface: Serif, Sans, Sans Narrow, and Mono. In each case there is a Regular (roman), Italic, Bold, and Bold Italic variant. \TeX lets you specify the type (extension) of the font, where it is installed, and how the font files names fit the pattern:

```
\setmainfont{LiberationSerif}[Extension=.ttf,  
Path=/home/peter/texmf/fonts/truetype/liberation/,  
UprightFont=*-Regular,  
BoldFont=*-Bold,  
ItalicFont=*-Italic,  
BoldItalicFont=*-BoldItalic]
```
The asterisk gets replaced by the font family name you give in the first argument, and the filetype (extension) is added to the end, so that \texttt{XELATEX} can construct the whole font name. Repeating this for the sans and mono variants gives us the full set (the Narrow face is an exception and less commonly used):

\begin{verbatim}
\setfamily{LiberationSans}[\sffamily][\mathversion{bold}][\mathversion{italic}]
\end{verbatim}

These now automatically fill the \LaTeX roles of the \texttt{rm}, \texttt{sf}, and \texttt{tt} families. This method can only be used with \texttt{XELATEX}.

If you are mixing OpenType or TrueType fonts from different families, there is a useful option keyword \texttt{Scale} which can be set as \texttt{Scale=}\texttt{MatchLowercase} for the sans and mono setups, making them scale to match the roman face.

The documentation for the \texttt{fontspec} package is very extensive and provides a lot of other facilities for managing TT and OT fonts.

### 6.2.3 Changing the default font family with \LaTeX and PDF\LaTeX

To set a typeface as the default family for a whole document, use the relevant package (see Table 6.2 to Table 6.5 on pages 142–146). For example, to set your whole document in New Century Schoolbook:

\begin{verbatim}
\usepackage[utf8x]{inputenc}
\usepackage[T1]{fontenc}
...
\usepackage{newcent}
\end{verbatim}

In these tables:

- \textbf{The full name is shown on the left-hand side.}
<table>
<thead>
<tr>
<th>Typeface Name</th>
<th>Font Family</th>
<th>Example Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer Modern Roman</td>
<td>cmr</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Computer Modern Sans</td>
<td>cmss</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Computer Modern Typewriter</td>
<td>cmtt</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Pandora (pandora) by Neenie Billawala</td>
<td>panr</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Pandora Sans</td>
<td>pss</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Pandora Typewriter</td>
<td>pntt</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Universal (uni) by Herbert Bayer</td>
<td>uni</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Concrete (concrete)</td>
<td>ccr</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Eireannach (eiad) by Ivan A Derzhanski</td>
<td>eiad</td>
<td>Éireannach (eiad) by Ivan A Derzhanski</td>
</tr>
<tr>
<td>Rustic (rustic) by Peter R Wilson</td>
<td>rust</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Uncial (uncial) by Peter R Wilson</td>
<td>uncl</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Dürer (duerer) by Alan Hoenig</td>
<td>cdr</td>
<td>Typographia Ars Artium Omnium Conservatrix</td>
</tr>
<tr>
<td>Fraktur (oldgerm) by Yannis Haralambous</td>
<td>U/yfrak</td>
<td>Fuěņ, Du hać die Hände gekloven, gib sie wieder her!</td>
</tr>
<tr>
<td>Gothic</td>
<td>U/ygoth</td>
<td>The quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Schwäbische</td>
<td>U/yswab</td>
<td>Suchs, Du hać die Hände gekloven, gib sie wieder her!</td>
</tr>
</tbody>
</table>
6.2. USING FONTS

If it is available as a package, the package name is given in parentheses. Fonts with mathematical symbols available are prefixed with an asterisk.

The short fontname (used in the \fontfamily command) is shown on the right-hand side.

If a non-standard font encoding is needed, it is shown as a prefix to the fontname, separated by a slash. Font encodings are ways of translating the font designer’s file layout into the Adobe one expected by Postscript and PDF processors.

Table 6.3: The Adobe '35': 10 Latin-alphabet typefaces (35 fonts in PostScript Type 1 format)

<table>
<thead>
<tr>
<th>Font Family</th>
<th>Fontname</th>
<th>Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avant Garde</td>
<td>avant</td>
<td>pag</td>
</tr>
<tr>
<td>Bookman</td>
<td>bookman</td>
<td>pbk</td>
</tr>
<tr>
<td>Courier</td>
<td>courier</td>
<td>pcr</td>
</tr>
<tr>
<td>Helvetica</td>
<td>helvet</td>
<td>phv</td>
</tr>
<tr>
<td>New Century Schoolbook</td>
<td>newcent</td>
<td>pnc</td>
</tr>
<tr>
<td>Palatino</td>
<td>mathpazo</td>
<td>ppl</td>
</tr>
<tr>
<td>Symbol</td>
<td>U/psy</td>
<td></td>
</tr>
<tr>
<td>Της θυίκη βρων ϕος ϕωμπο σαφη της λαζψ δογ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Times Roman</td>
<td>mathptmx</td>
<td>ptm</td>
</tr>
<tr>
<td>Zapf Chancery</td>
<td>chancery</td>
<td>pzc</td>
</tr>
<tr>
<td>Zapf Dingbats</td>
<td>pifont</td>
<td>U/pzd</td>
</tr>
</tbody>
</table>

*PostScript* Type 1 fonts were the mainstay of the graphic arts industries for many years, as they allowed very fine definition of variance (‘hinting’) compared with other formats. Despite the advances...
CHAPTER 6. LAYOUTS AND FONTS

with OpenType, TrueType, FreeType, and other formats, there are probably still millions if not billions of PostScript fonts installed worldwide. However, the font format remains proprietary to Adobe, even though they have released it for public use, which means that it could change without warning. OpenType was designed to overcome this risk, hence the use of OpenType fonts in XeLaTeX.

Table 6.4: Latin-alphabet fonts (PostScript Type 1) from the X Consortium

<table>
<thead>
<tr>
<th>Font Family</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charter (charter)</td>
<td>bch</td>
</tr>
<tr>
<td>The quick brown fox jumps over the lazy dog</td>
<td></td>
</tr>
<tr>
<td>Nimbus Roman (nimbus)</td>
<td>unm</td>
</tr>
<tr>
<td>The quick brown fox jumps over the lazy dog</td>
<td></td>
</tr>
<tr>
<td>Nimbus Sans</td>
<td>unms</td>
</tr>
<tr>
<td>The quick brown fox jumps over the lazy dog</td>
<td></td>
</tr>
<tr>
<td>URW Antiqua Condensed (urw)</td>
<td>uaq</td>
</tr>
<tr>
<td>The quick brown fox jumps over the lazy dog</td>
<td></td>
</tr>
<tr>
<td>URW Grotesk</td>
<td>ugq</td>
</tr>
<tr>
<td>The quick brown fox jumps over the lazy dog</td>
<td></td>
</tr>
<tr>
<td>Utopia (utopia)</td>
<td>put</td>
</tr>
<tr>
<td>The quick brown fox jumps over the lazy dog</td>
<td></td>
</tr>
</tbody>
</table>

You should note that these are just the defaults installed with all full versions of L\TeX fonts. There are hundreds more listed in Palle Jørgensen’s comprehensive \TeX Font Catalog published by the Danish \TeX Users Group at www.tug.dk/FontCatalogue/, categorised by type (serif, sans, monospace, decorative, etc) with samples and links to the packages on CTAN so that you can install them. However, not all of the zip files are yet in TDS, so refer to Table 3.1 on page 63 to make sure you put the files in the right places.

\TeX expects to work with three font families available all the time:

<table>
<thead>
<tr>
<th>Font family</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roman (serif, with tails on the uprights), the default</td>
<td>rm</td>
</tr>
<tr>
<td>Sans-serif, with no tails on the uprights</td>
<td>sf</td>
</tr>
<tr>
<td>Monospace (fixed-width or typewriter)</td>
<td>tt</td>
</tr>
</tbody>
</table>

The default for \TeX equates the rm with the \texttt{cmr} font-family (Computer Modern Roman), the sf with the \texttt{cmss} font-family (Computer
Modern Sans-Serif), and the \texttt{tt} with the \texttt{cmtt} font family (Computer Modern Typewriter).

As it is common to want to change all three defaults at the same time, a few of the most popular typeface packages change the default sans-serif and monospace typefaces automatically to make suitable companions (see below), but they are for text work only, as they leave any mathematics in Computer Modern. In these cases you may want to change the new default sans-serif or monospace typeface independently of the roman typeface if you need something different.

\texttt{times} changes to Times/Helvetica/Courier.

\texttt{pslatex} same as \texttt{times} but uses a specially narrowed Courier to save space (normal Courier is rather inelegantly wide). This is the preferred setting if you want Times without mathematics.\footnote{The \texttt{pslatex} package is also said to be outdated by some experts because it implements rather long-windedly what can now be done in three commands. However, until these replace the current version, I recommend continuing to use \texttt{pslatex} when you want Times with Helvetica and narrow Courier.}

\texttt{newcent} changes to New Century Schoolbook/Helvetica/Courier.

\texttt{palatino} changes to Palatino/Avant Garde/Courier.

The Helvetica typeface family has a notoriously large x-height (remember Figure 1.6 on page 25?), making it hard to match with other typefaces at the same nominal size. The \texttt{helvet} package therefore has a \texttt{scaled} option that lets you reduce the optical size slightly so that the font fits more easily with others: \texttt{\usepackage[\texttt{scaled}=0.86]\{helvet\}}, for example.

If you use mathematics, there are two fairly complete implementations of non-CM typefaces in the \texttt{mathptmx} (Times) and \texttt{mathpazo} (Palatino) packages. The whole business of math fonts is perpetually under revision in any case, as mathematicians are constantly inventing new symbols, which take a while to appear in typefaces. The American Mathematical Society (AMS) and other organisations are involved with a project called Stix, which is expected eventually to define a complete suite of mathematical characters in a rational and extensible manner — but don’t hold your breath.
CHAPTER 6. LAYOUTS AND FONTS

Table 6.5: Selection of non-Latin-alphabet typefaces (METAFONT)

<table>
<thead>
<tr>
<th>Font</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cypriot</td>
<td>cypriot</td>
</tr>
<tr>
<td>Etruscan</td>
<td>etruscan</td>
</tr>
<tr>
<td>Linear 'B'</td>
<td>linearb</td>
</tr>
<tr>
<td>Phoenician</td>
<td>phoenician</td>
</tr>
<tr>
<td>Runic</td>
<td>runic</td>
</tr>
<tr>
<td>Bard</td>
<td>U/zba</td>
</tr>
</tbody>
</table>

6.2.4 Changing the font-family temporarily with \texttt{\LaTeX}

To add a new font family, use the command \texttt{\newfontfamily}, which works exactly the same as the commands for setting the main, sans, and mono font families but needs an extra parameter first, to specify the command you want it known by:

\begin{verbatim}
\newfontfamily{\tablesfont}{LiberationSansNarrow}[Extension=.ttf, Path=/home/peter/texmf/fonts/truetype/liberation/, UprightFont=\*-Regular, BoldFont=\*-Bold, ItalicFont=\*-Italic, BoldItalicFont=\*-BoldItalic]
\end{verbatim}

Then the new command (here, \texttt{\tablesfont}) can be used to switch to that typeface.

To load a solitary font (that is, not a whole family), there is a command \texttt{\newfontface}, which also works in the same way, by creating a new command to switch to it.
6.2. USING FONTS

\newfontfamily{headlinefont}[LobsterTwo-Bold]
[Extension=.otf,Path=/usr/share/fonts/opentype/lobstertwo/]
...
{\centering\headlinefont\fontsize{20}{24}\selectfont
Lobster Rolls $3.95\par}

Lobster Rolls $3.95

Bear in mind that these new commands are unscoped, so they need to be inside a group (enclosed in curly braces as in the example, or within an environment), otherwise they will apply to the end of the document. See the panel ‘Grouping’ on p. 148 for more detail.

See section 6.2.5 on page 147 if you want to switch from an OpenType/TrueType font to a METAFONT or Type 1 font or family, as you will need to change the encoding.

6.2.5 Changing the font-family temporarily with \LaTeX and PDF\LaTeX

To shift to another font family on a temporary basis, there are commands for setting the font encoding and font family. You MUST end these commands with the command \selectfont, which is the trigger to implement them (exactly like you use \maketitle to trigger the creation of the title page after setting up the values).

Font encoding

In X\LaTeX, the default font encoding is EU1, which is inherent to OpenType and TrueType fonts. You will need to use \fontencoding if you want to switch to a non-OTF/TTF font within a X\LaTeX document.

If you are using plain \LaTeX or PDF\LaTeX the default font encoding SHOULD be set to T1 as we saw in step 2 on page 41. You will only need to use a \fontencoding command if you want to switch to a font with a different encoding (see the tables in section 6.2 on page 137).
CHAPTER 6. LAYOUTS AND FONTS

As all the common text fonts that come with \LaTeX are in T1 encoding, most of the time all you need is the \fontfamily and \selectfont commands:

\begin{verbatim}
{\fontfamily{phv}\selectfont Helvetica looks like this};\ \\
{\fontfamily{bch}\selectfont Charter looks like this}.
\end{verbatim}

- Helvetica looks like this;
- Charter looks like this.

These commands, like the ones in section 6.2.4 on page 146, are called \textit{unscoped} because they have global effect from that point on. In order to restrict the effect to a smaller scope (a few words, for example), you \textbf{MUST} use both the commands \textit{and} the text inside a \texttt{group} (enclosed in curly braces as in the example, or within an environment), otherwise they will apply to the end of the document. See the panel ‘Grouping’ on p. 148 for more detail.

\section*{Grouping}

This is a different way of using curly braces to how we have used them before — it limits the effect of simple unscoped commands to the material inside the braces, so it won’t ‘leak out’ and affect what follows. This use of curly braces therefore restricts the scope of a typographic change, instead of delimiting the argument to a command. This is called a \TeX \textit{group}. Inside a group, the effect of any changes is local, so they will not interfere with the text following the closing curly-brace.

If you use a paragraph-formatting command like \texttt{centering}, \texttt{flushleft}, or \texttt{flushright} inside a group, you \textbf{MUST} end the text with a \texttt{par} command \textit{inside the group} to cause the paragraph to be typeset with the desired format, otherwise the formatting simply will not take effect.

Environments like \texttt{center}, \texttt{quotation}, \texttt{table}, or \texttt{figure} are also groups, so the same rules apply, except that you do \textit{not} need the \texttt{par} at the end because most environments are inherently paragraph-based and will do it for you.
In a normal document, of course, arbitrary typeface changes like this are rare: people don’t (or at least, shouldn’t) randomly flip from one font to another. You select your default typefaces once, using packages or commands, at the start of the document, and stick with them — bold and italics are handled by the document class or stylesheet packages you use.

However, in advertising or magazines, a wide range of typefaces changes is common, but they are usually part of predefined commands for handling that type of formatting, built into the document class, so it is rare to have to do them manually.

Most cases where people want unusual typeface changes involve things like special symbols or effects on a repetitive basis, and \LaTeX\ provides much easier (programmable) ways to make these changes into shorthand commands (called macros: see Chapter 7 starting on page 161).

Vastly more common are changes to type \textit{style}, while staying within the same font-family.

### 6.2.6 Changing type style

Within each typeface or font family there are usually several different ‘looks’ to the type design. \LaTeX\ distinguishes mainly between \textit{font shape} and \textit{font series}. \textit{Italics} is a shape (look carefully: the actual shape of the letters changes, as well as their slope); whereas bold is a series (same shapes, same slope, just thicker strokes).

<table>
<thead>
<tr>
<th>Type style</th>
<th>Command</th>
<th>Example (using Computer Modern)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upright</td>
<td>\upshape</td>
<td>Quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Italic</td>
<td>\itshape</td>
<td>Quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Slanted</td>
<td>\slshape</td>
<td>Quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Small Caps</td>
<td>\scshape</td>
<td>Quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Bold</td>
<td>\bfseries</td>
<td>Quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Extended</td>
<td>\bfsseries</td>
<td>Quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Sans-serif</td>
<td>\sffamily</td>
<td>Quick brown fox jumps over the lazy dog</td>
</tr>
<tr>
<td>Monospace</td>
<td>\ttfamily</td>
<td>Quick brown fox jumps over lazy dog</td>
</tr>
</tbody>
</table>

`Formatting Information`
CHAPTER 6. LAYOUTS AND FONTS

The ‘shape’, ‘series’, and ‘family’ commands in Table 6.6 on page 149 are *commutative*, so you can combine a shape with a series and/or a family, without the need to use \selectfont:

```
This is (\bfseries\itshape\sffamily bold italic sans-serif type).
```

This is bold italic sans-serif type.

Beware of pushing your fonts beyond their limits unless you are a typographer. It is not normally meaningful to combine one shape or series class with another of the same class, such as trying to get slanted-italics. It’s also an impossibility to combine one family with another (such as seriffed sans-serif type!). Slanted plus italics, for example, doesn’t make sense, as italics are already slanted; and while some typefaces may well possess sans-serif italic small caps, they are not in common use. Sans-serif and monospace (typewriter) are not just different fonts, they are usually different typeface families entirely.\footnote{Although if you’re a typographer wanting to experiment with typewriter typefaces with and without serifs, you can use METAFONT or FontForge to do exactly this kind of thing. But that’s way outside the scope of this document.}

To avoid the problem of forgetting to put curly braces around the commands and text you want formatted, there is an alternative set of **scoped** commands for the most common type shape and series commands. These use curly braces in the ‘argument’ manner, so their effect applies only to the text in curly braces. These are the normal commands for changing the style of a word or phrase.

**Table 6.7:** Typeface styles, families, shapes, and series (scoped)

<table>
<thead>
<tr>
<th>Type style</th>
<th>Command</th>
<th>Example (using Computer Modern)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Italic</td>
<td>\textit{text}</td>
<td>puts text into italics</td>
</tr>
<tr>
<td>Slanted</td>
<td>\textsl{text}</td>
<td>puts text into slanted type*</td>
</tr>
<tr>
<td>Small Capitals</td>
<td>\textsc{text}</td>
<td>puts TEXT into small caps</td>
</tr>
<tr>
<td>Bold</td>
<td>\textbf{text}</td>
<td>puts text into bold type</td>
</tr>
<tr>
<td>Sans-serif</td>
<td>\textsf{text}</td>
<td>puts text into sans-serif type</td>
</tr>
<tr>
<td>Monospace</td>
<td>\texttt{text}</td>
<td>puts text into typewriter type</td>
</tr>
</tbody>
</table>

These are commutative too, so you can nest them inside one another:
6.2. USING FONTS

What we know as **underlining** isn’t a font: it was used in the days of typewriters where italics were not available, and it is extremely rare in typography except for specialist purposes. If you think you need it, use the **ulem** package with the **normal em** option, and the **\uline** command.

6.2.7 Font sizes

\TeX{} has built into its defaults a set of predefined font size steps corresponding more or less to the traditional sizes available to metal typesetters. This is deliberate, as these sizes have grown up over 500 years of experience in printing as those which go best together for book-work, which is where \TeX{} originated.

These sizes are also reflected in the **size steps** at which Computer Modern was designed in the METAFONT program. It often comes as a surprise to new users that many typefaces are not designed as a single font and just scaled up or down, but specially drawn at different sizes to make them more legible.

As an example, here’s 12pt Computer Modern, and here’s 5pt Computer Modern scaled up to 12pt, and here’s 17pt Computer Modern scaled down to 12pt so you can see there really is a significant difference. Type 1 and other font formats have **hinting** parameters that allow some scaling to implement the effects of design-sizes, but in general, you probably don’t want to go scaling fonts too far beyond their design size because the spacing will start to look very odd.

The default sizes (and the commands that operate them) are based on the use of a 10pt font, which is the default size for book work. Using the larger defaults (11pt and 12pt) for the body font will use 11pt and 12pt designs, with other sizes (such as for headings) rescaled to match. The exact sizes used are listed in the macros in the Class Option files **size10.clo**, **size11.clo** and **size12.clo**. \TeX{}’s default fonts above 10pt are in fact scaled by a factor of 1.2, as shown in the fourth column of Table 6.8 on page 152.

While these shorthand commands relieve the beginner of having to worry about the ‘right’ size for a given task, if you need very...
CHAPTER 6. LAYOUTS AND FONTS

Table 6.8: \LaTeX{} font step sizes

<table>
<thead>
<tr>
<th>Command</th>
<th>Example</th>
<th>Nominal point size</th>
<th>Exact point size</th>
</tr>
</thead>
<tbody>
<tr>
<td>\tiny</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>\scriptsize</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>\footnotesize</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>\small</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>\normalsize</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>\large</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>\Large</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>14</td>
<td>14.40</td>
</tr>
<tr>
<td>\LARGE</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>18</td>
<td>17.28</td>
</tr>
<tr>
<td>\huge</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>20</td>
<td>20.74</td>
</tr>
<tr>
<td>\Huge</td>
<td>The quick brown fox jumps over the lazy dog</td>
<td>24</td>
<td>24.88</td>
</tr>
</tbody>
</table>

Note that these are unscoped commands, so they should be used inside a group, either an environment or a set of curly braces terminated with a \par inside the closing brace.

Specific sizes you can use the fix-cm package to override the step sizes. This needs special placement: it has to come at the start of the document, even before the \documentclass command, and must be invoked with the \RequirePackage command normally used only inside packages:

```latex
\RequirePackage{fix-cm}
\documentclass[a4paper,12pt]{article}
```

You can then use the \fontsize command to specify exact sizes. This takes two arguments: the point size and the baseline distance. The
example here gives you 22pt type on a 28pt baseline (ie with 6pt extra space or ‘leading’ between the lines).

\begin{verbatim}
\fontsize{22}{28}\selectfont
The example here gives you 22pt type 6pt leaded (that is, with 6pt extra space between the lines, making a line-height of 28pt). \par
\end{verbatim}

The example here gives you 22pt type 6pt leaded (that is, with 6pt extra space between the lines, making a line-height of 28pt).

‘Leading’ comes from the old metal-type practice of adding a strip of typemetal between lines to increase the spacing (so it’s pronounced ‘ledding’ after the metal).

### 6.2.8 Logical markup

All this playing around with fonts is very pretty but you normally only do it for a reason, even if that reason is just to be decorative. Italics, for example, are used for many things:

<table>
<thead>
<tr>
<th>Cause</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreign words</td>
<td>ex officio</td>
</tr>
<tr>
<td>Scientific names</td>
<td>Ranunculus ficaria</td>
</tr>
<tr>
<td>Emphasis</td>
<td>must not</td>
</tr>
<tr>
<td>Titles of documents</td>
<td>Accounting in Business</td>
</tr>
<tr>
<td>Product names</td>
<td>Corel WordPerfect</td>
</tr>
<tr>
<td>Variables in maths</td>
<td>$E = mc^2$</td>
</tr>
<tr>
<td>Subtitles or headings</td>
<td>42. How to get started</td>
</tr>
<tr>
<td>Use of a letter as a word</td>
<td>Who knocked the L out of London?</td>
</tr>
<tr>
<td>Decoration</td>
<td>FREE UPGRADE!!</td>
</tr>
</tbody>
</table>

"Formatting Information"
Humans usually have no problem telling the difference between these reasons, because they can read and understand the meaning and context. Computers cannot (yet), so it has become conventional to use descriptive names which make the distinction explicit, even though the appearance may be the same.

\TeX{} has some of these built in, like \texttt{\textitshape}, which provides emphasis. This has a special feature because when the surrounding text is already italic, emphasis automatically reverts to upright type, which is the normal practice for typesetting.

This sensitivity to logic is programmed into the definition of \texttt{\textitshape} and it's not hard to make up other commands of your own which could do the same, such as \texttt{\textforeign} or \texttt{\textproduct}.

But why would you bother? In a short document it's probably not important, but if you're writing a long report, or a formal document like an article, a book, or a thesis, it makes writing and editing hugely easier if you can control whole groups of special effects with a single command, such as italicising, indexing, or cross-referencing to a glossary. If a format needs changing, you only have to change the definition, and every occurrence automatically follows suit.

\begin{quote}
A warning from the past

Beware of this 'vaine conceipt of simple men, which judge things by ther effects, and not by ther causes'. (Edmund Spenser, 1633)

It's hugely more efficient and productive to have control of the cause than the effect.

It also makes it possible to find and act on groups of meanings — such as making an index of scientific names, or retrieving all product names — if they are identified as such. Otherwise you'd spend weeks hunting manually through every \texttt{\textit} command to find the ones you wanted. This is the bottom line of automation: it can save you time and money.
\end{quote}
In Chapter 7 starting on page 161 we will see how to make your own simple commands to do things like this.

### 6.2.9 Colour

You can typeset anything in \LaTeX\ in any colour you want using the xcolor package. Adding the command `\usepackage{xcolor}` to your Preamble (note the US spelling of color) makes available a default palette of primary colours: red, green, and blue for the RGB colour model used for emitted light (computer and television screens), and cyan, magenta, yellow, and black for the CMYK colour model used for reflected light (printing).

For the occasional word or phrase in colour, use the command `\textcolor` with two arguments, the colour name and the text: `\textcolor{red}{like this}` to get red like this. There is an unscoped `\color` command as well, for use within groups:

```
...{\color{blue}some text in blue}...
```

There are several package options for additional colours: two popular ones are dvipsnames, which provides a 64-colour palette of predefined colour names matching the big box of 64 Crayola™ colouring pencils much favoured by artists and designers; and svgnames, which provides the 256 colours defined in the specification for the Scalable Vector Graphics (SVG) drawing and diagramming language. There are others too: see the documentation for the xcolor package.

If you want the Crayola colour Crimson, and you have loaded xcolor with the dvipsnames option, you can use it as a colour name:

```
{\color{Crimson}some red text}
\textcolor{Crimson}{some red text}
```

As some of the predefined colour names are quite long, you can create a short name of your own for colours you use frequently, using the `\definecolor` command:

```
\definecolor{mb}{named}{MidnightBlue}
```

The `\definecolor` command needs three arguments: your shorthand name, the name of the colour model, and the colour specification.
In the case of the \texttt{named} model, the last argument is one of the colour names specified by the named option you loaded the package with.

Using the \texttt{\definecolor} command, you can also define any colour you want by giving it a name, specifying which colour model, and providing the Red-Green-Blue (RGB) or Cyan-Magenta-Yellow-Black (CMYK) colour values \textit{expressed as decimal fractions of 255, separated by commas}. For example, an RGB colour given as (37,125,224) in decimal integer form can be given as:

\begin{verbatim}
\definecolor{midblue}{rgb}{0.145,0.490,0.882}
\end{verbatim}

To get the fractional value, divide the integer value by 255, the maximum for each of the hues in the Red-Green-Blue colour model. You can then use \texttt{\textcolor} with your new colour name: \texttt{\textcolor{midblue}{} looks like this} if you're reading in colour. Alternatively, use the HTML hexadecimal colour model, the same as used in web pages and CSS stylesheets.

\begin{verbatim}
\definecolor{midblue}{HTML}{250FE0}
\end{verbatim}

The \texttt{xcolor} package also provides two colour versions of \texttt{\fbox} (see section 4.6.2 on page 101) called \texttt{\colorbox} and \texttt{\fcolorbox} which create a box with a coloured background:

\begin{verbatim}
\colorbox{midblue}{\color{magenta}Magenta on midblue}
\end{verbatim}

The material in the second argument can have its own text colour, as in the example. The \texttt{\fcolorbox} has an extra first argument to specify the colour of the frame or border placed around the box. The border width is controlled by the \texttt{\fboxrule} setting and the separation between rule and content is controlled by \texttt{\fboxsep} setting as we already saw in section 4.6.2 on page 101.

However, combining colours is an art and a skill: using the command \begin{verbatim}
\colorbox{midblue}{\color{magenta}magenta on midblue}
\end{verbatim} to get the effect \texttt{magenta on midblue} illustrates why it is important to learn about colour models and palettes before trying to use them!
6.3 The \LaTeX{} font catalogue

The \LaTeX{} Font Catalog is a web site created and maintained by Palle Jørgensen at \url{www.tug.dk/FontCatalogue/}. It lists over 200 typefaces for use with \LaTeX{}, many of them available nowhere else, with samples and links to the directories on CTAN where you can download them. You can spend many fascinating hours downloading and installing them and trying them out in your documents.

For newcomers, installing a new typeface can appear challenging, when described as I do for the manual installation of PostScript fonts in Appendix 2 starting on page 229. But the typefaces in the \LaTeX{} Font Catalog are prebuilt for \LaTeX{}, so all you have to do is download the \texttt{.zip} file, unzip it into your personal \TeX{} directory, and move the subdirectories into the right places. A worked example is the best way to describe this.

Let’s suppose we want to install the sans-serif Kurier typeface. This is nothing to do with the Courier typewriter face, but was designed in pre-computing times by Małgorzata Budyta, and digitised and extended by Janusz M Nowacki (thanks to the GUST web site for this information).

Installing a font from the Font Catalogue

1. If we click on the name in the sans-serif page of the Catalog, we can see a sample paragraph, and we can click on the link at the bottom of the page to go to the CTAN directory where the typeface is stored.

2. Here there is a brief README file, and links to the individual font subdirectories, but most importantly there is a link at the top of the page to the \texttt{.zip} file containing it all. Download this and unzip it straight into your personal \TeX{} directory (see section A.2 on page 217 for what this is and where to create it).

3. Now open your personal \TeX{} directory in your directory browser (Windows: \textit{My Computer} or just \textit{Computer}; Mac: \textit{Finder}; Linux: \textit{Nautilus}, \textit{Thunar}, or \textit{Dolphin}). You will see that inside the \texttt{kurier} directory there are subdirectories called \texttt{doc}, \texttt{fonts}, and \texttt{tex} (see Figure 6.2 on page 158).
CHAPTER 6. LAYOUTS AND FONTS

Figure 6.2: Layout of a font zip file downloaded from CTAN

Figure 6.3: Location of the map file for a typeface downloaded from CTAN

4. Drag and drop each of those subdirectories into your Personal \TeX\ directory (\texttt{texmf}). If directories with the same names already exist, your system should ask if you want the new ones merged with the existing ones: the answer is yes, so click OK. In some typefaces there may be more subdirectories than shown here for Kurier: do the same with them all.

5. The next step is to find the \texttt{.map} file that \LaTeX\ needs to set up the link between its short (‘Karl Berry’) font names and the long ones used by \texttt{PDF} and \texttt{PostScript} output. This should be in the \texttt{fonts/map/dvips/fontname} subdirectory of your personal \TeX\ directory (see Figure 6.3 on page 158). If not, use the \texttt{$kpsewhich} command in a Command window to find it, or use your system’s file-finder.
In this case there are many map files, but the one we want is just called \texttt{kurier.map} (the others are there in case you only wanted to install a single font, not the whole typeface).

6. The last step is to run the font map update program \texttt{updmap} to enable use of the map file. You need to do this in a terminal or command window, by typing the command

\begin{verbatim}
updmap --enable Map=kurier.map
\end{verbatim}

This reloads all your font maps, so it takes a minute or so to run. Once that’s done, you can \texttt{\usepackage{kurier}} in your documents and start using the typeface.
We’ve touched several times on the ability of \LaTeX{} to be reprogrammed. This is one of its central features, and one that still, after nearly a quarter of a century, puts it well above many other typesetting systems, even those with programming systems of their own. It’s also the one that needs most foreknowledge, which is why this chapter is in this position.

\TeX{} is basically a programming language for typesetting. As such, it allows you to define \textit{macros}, which are little (or large) program-like sequences of commands with a name which can be used as a command itself. This in effect makes a macro a shorthand for a sequence of operation you wish to perform more than once. \LaTeX{} is in fact just a large collection of such macros.

Macros can be arbitrarily complex. Many of the ones used in the standard \LaTeX{} packages are several pages long, but as we will see, even short one-liners can very simply automate otherwise tedious chores and allow the author to concentrate on the most important thing; \textit{writing}.

\section{Simple replacement macros}

In its simplest form, a \LaTeX{} macro can just be a straightforward text replacement of a phrase to avoid lengthy retyping with the possibility of misspelling something each time you need it, eg

\begin{verbatim}
\textbf{Formatting Information}
\end{verbatim}
\newcommand{\EF}{European Foundation for the Improvement of Living and Working Conditions}

Put this in your Preamble, and you can then use \EF in your document and it will typeset it as the full text. Remember that after a command ending in a letter you need to leave a space to avoid the next word getting gobbled up as part of the command (see section 1.5.1 on page 13). If you want to force a space to be printed after the expansion, use a backslash followed by a space, eg

The \EF\ is a member institution of the Commission of the European Union.

As you can see from this example, the \newcommand command takes two arguments: the name you want to give the new command, and the expansion to be performed when you use it, so there are always two sets of curly braces after a \newcommand. The names of new commands created like this MUST be made of the letters A–Z and a–z only, and must not be the names of existing commands.

### 7.2 Macros using information gathered previously

A more complex example is the macro \maketitle which is used in almost every \LaTeX document to format the title block. In the default document classes (book, report, and article) it performs small variations on the layout of a centred block with the title followed by the author followed by the date, as we saw in section 2.3 on page 42.

If you inspect one of the default document class files, such as report.cls you will see \maketitle defined (and several variants called \@maketitle for use in different circumstances). It uses the values for the title, author, and date which are assumed already to have been stored in the internal macros \@title, \@author, and \@date by the author using the matching \title, \author, and \date commands in the document before using the \maketitle command.

This use of one command (eg \title) to store the information in another (eg \@title) is a common way of gathering the information from the user. The use of macros containing the @ character prevents
7.2. MACROS USING INFORMATION GATHERED PREVIOUSLY

their accidental misuse by the user because they designed for use in packages and classes, and are disallowed in document text. To use them in your Preamble (eg to redefine \texttt{\maketitle}), you have to allow the @ sign to temporarily become a ‘letter’ using the \texttt{\makeatletter} command so the @ can be recognised in a command name (and remember to turn it off again afterwards with the \texttt{\makeatother} command — see item 1 on page 163 below).

\begin{verbatim}
\makeatletter
\renewcommand{\maketitle}{%
 \begin{flushleft}%
 sffamily
 \Large\bfseries\color{red}@title\par%
 \medskip
 \large\color{blue}@author\par%
 \medskip
 \itshape\color{green}@date\par%
 \bigskip\hrule\vspace*{2pc}%
 \end{flushleft}%
 }
\makeatother
\end{verbatim}

Insert this immediately before the \texttt{\begin{document}} in the sample file in the first listing, and make sure you have used the \texttt{xcolor} package. Typeset the file and you should get something like Figure 7.1 on page 164.

In this redefinition of \texttt{\maketitle}, we’ve done the following:

1. Enclosed the changes in \texttt{\makeatletter} and \texttt{\makeatother} to allow us to use the @ sign in command names;\footnote{If you move all this Preamble into a package (.sty) file of your own, you don’t need these commands: the use of @ signs in command names is allowed in package and class files.}

2. Used \texttt{\renewcommand} and put \texttt{\maketitle} in the first pair of curly braces after it;

3. Opened a second pair of curly braces to hold the new definition. The closing curly brace of this pair is immediately before the \texttt{\makeatother};
4. Inserted a **flushleft** environment so the whole title block is left-aligned;

5. Used `\sffamily` so the whole title block is in the defined sans-serif typeface;

6. For each of `@title`, `@author`, and `@date`, we have used some font variation and colour, and enclosed each one in curly braces to restrict the changes just to each command. The closing `\par` of each one makes sure that multiline title and authors and dates would get typeset with the relevant line-spacing;

7. Added some flexible space between the lines, and around the `\hrule` (horizontal rule) at the end;

Note the `%` signs after any line ending in a curly brace, to make sure no intrusive white-space find its way into the output. These aren’t needed after simple commands where there is no curly brace because excess white-space gets gobbled up there anyway.

Figure 7.1: Example of reprogrammed title layout
Ex e r c ise 16. R e writing the title

Add the code above to your test document (or create a new one), then add title, author, and date, and make your new title.

7.3 Macros with arguments

But macros are not limited to text expansion or the reproduction of previously-stored values. They can take arguments of their own, so you can define a command to do something with specific text you give it. This makes them much more powerful and generic, as you can write a macro to do something a certain way, and then use it hundreds of times with a different value each time.

We mentioned earlier (in section 6.2.8 on page 154) the idea of making new commands to put specific classes of words into certain fonts, such as \textit{foreign} or \textit{product}. Here's an example for a new command \tmproduct, which also indexes the product name and adds a trademark sign:

\newcommand{\tmproduct}[1][]{%
  \textit{#1}\texttrademark%
  \index{#1@\textit{#1}}%
}

If I now type \tmproduct{Velcro}, I get Velcro™ typeset, and if you look in the index, you'll find this page referenced under Velcro. Let's examine what this does:

1. The macro is specified as having one argument (that’s the [1] in the definition). This will be the product name you type in curly braces when you use \product. Macros can have up to nine arguments.

2. The expansion of the macro is contained in the second set of curly braces, spread over several lines (see item 5 on page 166 for why).

3. It prints the value of the first argument (that’s the #1) in italics, which is conventional for product names, and adds the \texttrademark command.
CHAPTER 7. PROGRAMMABILITY

4. Finally, it creates an index entry using the same value (#1), making sure that it's italicised in the index (see the list item 'Font changes' section 5.4.1 on page 123 to remind yourself of how indexing something in a different font works).

5. Typing this macro over several lines makes it easier for humans to read. I could just as easily have typed

\newcommand{\product}[1]{\textit{#1}\index{#1@\textit{#1}}}

but it wouldn’t have been as clear what I was doing.

One thing to notice is that to prevent unwanted spaces creeping into the output when \LaTeX reads the macro, I ended each line with a comment character (%). \LaTeX normally treats newlines as spaces when formatting (remember the list item '%' section 1.6.1 on page 16), so this stops the end of line being turned into an unwanted space when the macro is used. \LaTeX usually ignores spaces at the start of macro lines anyway, so indenting lines for readability is fine.

In section 1.9.2 on page 26 we mentioned the problem of frequent use of unbreakable text leading to poor justification or to hyphenation problems. A solution is to make a macro which puts the argument into a \texttt{mbox} with the appropriate font change, but precedes it all with a conditional \texttt{linebreak} which will make it more attractive to \TeX to start a new line.

\newcommand{\var}[1]{\textttfamily\texttt{linebreak}[3]\texttt{mbox}\texttt{ttfamily#1}}

This only works effectively if you have a reasonably wide setting and paragraphs long enough for the differences in spacing elsewhere to get hidden. If you have to do this in narrow journal columns, you may have to adjust wording and spacing by hand occasionally.

7.4 Nested macros

Here’s a slightly more complex example, where one macro calls another. It’s common in normal text to refer to people by their forename and surname (in that order), for example Donald Knuth,
7.4. NESTED MACROS

but to have them indexed as surname, forename. This pair of macros, \texttt{\textbackslash person} and \texttt{\textbackslash reindex}, automates that process to minimise typing and indexing.

\begin{verbatim}
\newcommand{\person}[1]{\reindex #1\sentinel}
def\reindex #1 #2\sentinel{\index{#2, #1}}
\end{verbatim}

1. The digit 1 in square brackets means that \texttt{\textbackslash person} has one argument, so you put the whole name in a single set of curly braces, eg \texttt{\person{Don Knuth}}.

2. The first thing the macro does is output \texttt{#1}, which is the value of what you typed, just as it stands, so the whole name gets typeset exactly as you typed it.

3. But then it uses a special feature of Plain \TeX macros (which use \texttt{\def} instead of \LaTeX's \texttt{\newcommand}): they too can have multiple arguments but you can separate them with other characters (here a space) to form a pattern which \TeX will recognise when reading the arguments.

In this example (\texttt{\reindex}) it’s expecting to see a string of characters (\texttt{#1}) followed by a space, followed by another string of characters (\texttt{#2}) followed by a dummy command (\texttt{\sentinel}). In effect this makes it a device for splitting a name into two halves on the space between them, so the two halves can be handled separately. The \texttt{\reindex} command can now read the two halves of the name separately.

4. The \texttt{\person} command invokes \texttt{\reindex} and follows it with the name you typed plus the dummy command \texttt{\sentinel} (which is just there to signal the end of the name). Because \texttt{\reindex} is expecting two arguments separated by a space and terminated by a \texttt{\sentinel}, it sees ‘Don’ and ‘Knuth’ as two separate arguments.

It can therefore output them using \texttt{\index} in reverse order, which is exactly what we want.

\footnote{Don’t try this at home alone, folks! This one is safe enough, but you should strictly avoid \texttt{\def} for a couple of years. Stick to \texttt{\newcommand} for now.}
A book or report with a large number of personal names to print and index could make significant use of this to allow them to be typed as \person{Leslie Lamport} and printed as Leslie Lamport, but have them indexed as ‘Lamport, Leslie’ with virtually no effort on the author’s part at all.

### 7.5 Macros and environments

As mentioned in section 4.6.3 on page 102, it is possible to define macros to capture text in an environment and reuse it afterwards. This avoids any features of the subsequent use affecting the formatting of the text.

One example of this uses the facilities of the fancybox package, which defines a variety of framed box commands to display your text, but they require a pre-formed box as their argument, so the package provides a special environment \texttt{Sbox} which ‘captures’ your text for use in these boxes.

Here we put the text in a \texttt{minipage} environment because we want to change the width; this occurs \texttt{inside} the \texttt{Sbox} environment so that it gets typeset into memory and stored in a box. It can then be ‘released’ afterwards with the command \texttt{TheSbox} as the argument of the \texttt{shadowbox} command (and in this example it has also been centred).
This text is formatted to the specifications of the minipage environment in which it occurs.

Having been typeset, it is held in the Sbox until it is needed, which is after the end of the minipage, where you can (for example) align it and put it in a special framed box.

\end{minipage}
\end{Sbox}
\begin{center}
\shadowbox{\TheSbox}
\end{center}

The point about this kind of construct is that it can be turned into an environment of your own, so you can reuse it wherever you need:

\begin{warning}
\begin{Sbox}\begin{minipage}{8cm}\end{minipage}\end{Sbox}\begin{center}\shadowbox{\TheSbox}\end{center}
CHAPTER 7. PROGRAMMABILITY

7.6 Reprogramming \LaTeX’s internals

\LaTeX’s internal macros can also be reprogrammed or even rewritten entirely, although doing this can require a considerable degree of expertise. Simple changes, however, are easily done.

Recall that \LaTeX’s default document structure for the Report document class uses Chapters as the main unit of text, whereas in reality most reports are divided into Sections, not Chapters (footnote 6 on page 47). The result of this is that if you start off your report with \texttt{\section{Introduction}}, it will print as

\begin{verbatim}
0.1 Introduction
\end{verbatim}

which is not at all what you want. The zero is the (missing) chapter number, because no chapter has been started. But this numbering is controlled by macros, and you can redefine them. In footnote 2 on page 78 we said that every counter automatically gets a related command beginning with ‘the’. In this case what we need to change is that command \texttt{\thesection} because the way the counters are set up makes it reproduce the value of the counter \texttt{section} (see section 4.1.6 on page 79) plus any higher-level value (eg \texttt{chapter}). It’s redefined afresh in each document class file, using the command \texttt{\renewcommand} (in this case in \texttt{report.cls}):

\begin{verbatim}
\renewcommand{\thesection}{\thechapter.\arabic{section}}
\end{verbatim}

You can see it invokes \texttt{\thechapter} (which is defined elsewhere to reproduce the value of the \texttt{chapter} counter), and it then prints a dot, followed by the Arabic value of the counter called \texttt{section} (that \texttt{c@} notation is \LaTeX’s internal way of referring to counters). You can redefine this in your Preamble to simply leave out the reference to chapters:

\begin{verbatim}
\renewcommand{\thesection}{\arabic{section}}
\end{verbatim}

I’ve used the more formal modern method of enclosing the command being redefined in curly braces. For largely irrelevant historical reasons these braces are often omitted in \LaTeX’s internal code (as you may have noticed in the example earlier). And I’ve also used the ‘public’ version of the \texttt{\arabic} command to output the value of
section (\LaTeX’s internals use a ‘private’ set of control sequences containing @-signs, designed to protect them against being changed accidentally).

Now the introduction to your report will start with:

1 Introduction

What’s important in making this type of modification is that you DO NOT alter the original document class file \texttt{report.cls} (ever): you just copy the command you need to change into your own document Preamble, or a private package file, and modify that instead. It will then override the default because it will get loaded after the document class.

7.6.1 Changing list item bullets

As mentioned earlier, here’s how to redefine a bullet for an itemized list, with a slight tweak:

\begin{verbatim}
\usepackage{bbding}
...
\renewcommand{\labelitemi}{%\raisebox{-0.25ex}{\PencilRight}}
\end{verbatim}

Here we use the \texttt{bbding} package which has a large selection of ‘dingbats’ or little icons, and we change the label for top-level itemized lists (\texttt{\labelitemi}, find these in any document class file) to make it print a right-pointing pencil (the names for the icons are in the \texttt{bbding} package documentation: see section 3.1.3 on page 59 for how to get it).

In this case, we are also using the \texttt{\raisebox} command within the redefinition because it turns out that the symbols in this font are positioned slightly too high for the typeface we’re using. The \texttt{\raisebox} command takes two arguments: the first is a dimension, how much to raise the object by (and a negative value means ‘lower’: there is no need for a separate \texttt{\lowerbox} command); and the second is the text you want to affect. Here, we are shifting the symbol down by \texttt{1/4ex} (see section 1.9.1 on page 23 for a list of dimensional units \LaTeX can use).
There are label item commands for each level of lists (1–4) which have command names ending in Roman numerals (i–iv) because of the rule that command names can only use letters. Thus to change the icon for the fourth-level list, modify \texttt{labelitemiv}. There is a vast number of symbols available: see \textit{The \TeX\ Symbol List} for a comprehensive list.
Compatibility

As we saw right at the start, \LaTeX{} uses plaintext files, so they can be read and written by any standard application that can open text files. This helps preserve your information over time, as the plaintext format cannot be obsoleted or hijacked by any manufacturer or sectoral interest, and it will always be readable on any computer, from your smartphone (\LaTeX{} is available for many handhelds, from old PDAs, see Figure 8.1 on page 173, to Android devices, see Figure 8.2 on page 174) through all desktops and servers right up to the biggest supercomputers.

Figure 8.1: \LaTeX{} editing and processing on the Sharp Zaurus 5500 PDA
CHAPTER 8. COMPATIBILITY

Figure 8.2: \LaTeX{} editing and processing on the Samsung Galaxy Note 4

However, \LaTeX{} is intended as the last stage of the editorial process: formatting for print or display. If you have a requirement to re-use the text in some other environment — a database perhaps, or on the Web or a CD-ROM or DVD, or in Braille or voice output — then it should probably be edited, stored, and maintained in something neutral like the Extensible Markup Language (XML), and only converted to \LaTeX{} when a typeset copy is needed.

Although \LaTeX{} has many structured-document features in common with SGML and XML, it can still only be processed by the \LaTeX{}, PDF\LaTeX{}, and \XMLLaTeX{} programs. Because its macro features make it almost infinitely redefinable, processing it requires a program which can unravel arbitrarily complex macros, and \LaTeX{} and its siblings are the only programs which can do that effectively. Like other typesetters and formatters (Quark XPress, Adobe InDesign and PageMaker, FrameMaker, Microsoft Publisher, 3\textsc{b}2, etc), \LaTeX{} is largely a one-way street leading to typeset printing or display formatting.

Converting \LaTeX{} to some other format therefore means you will unavoidably lose some formatting, as \LaTeX{} has features that others systems simply don’t possess, so they cannot be translated — although there are several ways to minimise this loss or compensate for it. Similarly, converting other formats into \LaTeX{} often means editing back
the stuff the other formats omit because they only store appearances, not structure.

Most converters are one-way: that is, they convert into \LaTeX{} or out of \LaTeX{}, and there are several excellent systems for doing the conversion from \LaTeX{} directly to HyperText Markup Language (HTML) so you can at least publish it on the web, as we shall see in section 8.2.

However, there is one system that does both, and includes a huge range of formats: Pandoc (pandoc.org/). This is a large library of Haskell routines for handling conversions, with a command-line front end. Supported formats include Word, OpenOffice/Libre Office, DocBook, InDesign, MarkDown, and MediaWiki. Before trying the systems described in section 8.1 on page 175 and section 8.2 on page 182, see if Pandoc will handle your files. The exception is probably converting from XML to \LaTeX{} for which a robust XSLT2 script is really the only reliable solution.

8.1 Converting into \LaTeX{}

Before looking at one-way systems, see the earlier note about Pandoc.

There are several systems which will save their text in \LaTeX{} format. The best known is probably \LaTeX X, which is a wordprocessor-like interface to \LaTeX{} (not quite \textsc{wysiwyg}, more \textsc{what you see is what you mean}). Both \textsc{AbiWord} (Linux and Windows) and \textsc{Kword} (Linux) have a very good \textit{Save As…} \LaTeX{} output, and \textit{OpenOffice} (all platforms) has a \LaTeX{} plugin, so they can be used to open Microsoft \textit{Word} documents as well as their own format, and convert them to \LaTeX{}. Several maths packages like the \textit{EuroMath} editor, and the \textit{Mathematica} and \textit{Maple} analysis packages, can also save material in \LaTeX{} format.

In general, most other wordprocessors and DTP systems either don’t have the level of internal markup sophistication needed to create a \LaTeX{} file, or they lack a suitable filter to enable them to output what they do have. Often they are incapable of outputting any kind of structured document, because they only store what the text looks like, not why it’s there or what role it fulfills. There are two ways out of this:

- Use the \textit{File} → \textit{Save As…} menu item to save the wordprocessor file as HTML, rationalise the HTML using Dave Raggett’s \textit{HTML Tidy}, and
convert the resulting XHTML file to \LaTeX\ with any of the standard XML tools (see below).

\begin{itemize}
\item Get the files into Word or ODF format, and write a transformation in XSLT to convert the internal XML into \LaTeX. This is by far the most robust way to do it, but the quality of most wordprocessing files is poor when it comes to identifying which bits do what, which is what \LaTeX\ needs, so some guesswork or heuristics may be needed.
\end{itemize}

If you have large numbers of obsolete Word .doc files (too many to open and save as .docx), you can try to use a specialist conversion tool like EBT’s DynaTag (supposedly available from Enigma, if you can persuade them they have a copy to sell you; or you may still be able to get it from Red Bridge Interactive in Providence, RI). It’s old and expensive and they don’t advertise it, but for GUI-driven bulk conversion of consistently-marked Word (.doc, not .docx) files into usable XML it beats everything else hands down. But whatever system you use, the Word files **must** be consistent, though, and **must** use Named Styles from a stylesheet (template), otherwise no system on earth is going to be able to guess what they mean.

There is of course a fourth way, suitable for large volumes only: send it off to the Pacific Rim to be scanned or retyped into XML or \LaTeX. There are hundreds of companies from India to Polynesia who do this at high speed and low cost with very high accuracy. It sounds crazy when the document is already in electronic form, but it’s a good example of the problem of low quality of wordprocessor markup that this solution exists at all.

You will have noticed that most of the solutions lead to one place: XML. As explained above and elsewhere, this format is the only one so far devised capable of storing sufficient information in machine-processable, publicly-accessible form to enable your document to be recreated in multiple output formats. Once your document is in XML, there is a large range of software available to turn it into other formats, including \LaTeX. Processors in any of the common XML processing languages like XSLT or Omnimark can easily be written to output \LaTeX, and this approach is extremely common.
8.1. CONVERTING INTO \LaTeX

Much of this would be simplified if wordprocessors supported native, arbitrary XML/XSLT as a standard feature, because \LaTeX output would become much simpler to produce, but this seems unlikely.

However, the native format for both OpenOffice and Word is now XML. Both .docx and .odf files are actually Zip files containing the XML document together with stylesheets, images, and other ancillary files. This means that for a robust transformation into \LaTeX, you just need to write an XSLT stylesheet to do the job — non-trivial, as the XML formats used are extremely complex, but certainly possible.

Among the conversion programs for related formats on CTAN is Ujwal Sathyam and Paul DuBois’s rtf2latex2e, which converts Rich Text Format (RTF) files (output by many wordprocessors) to \LaTeX. The package description says it has support for figures and tables; equations are read as figures; and it can handle the latest RTF versions from Microsoft Word 97/98/2000, StarOffice, and other wordprocessors. It runs on Macs, Linux, other Unix systems, and Windows.

8.1.1 Getting \LaTeX out of XML

Assuming you can get your document out of its wordprocessor format into XML by some method, here is a very brief example of how to turn it into \LaTeX.

You can of course buy any fully-fledged commercial XML editor with XSLT support, and run transformations within it. However, this is beyond the reach of many individual users, although oXygen is available at a discounted price to academic sites.

To do the job unaided you need to install three pieces of software: Java, Saxon or another XSLT processor, and the DocBook 5.0 DTD (links are correct at the time of writing). None of these has a graphical interface: they are run from the command-line.

As an example, let’s take the above paragraph, as typed or imported into AbiWord (see Figure 8.3 on page 178). This is stored as a single paragraph with highlighting on the product names (italics), and the names are also links to their Internet sources, just as they are in this document. This is a convenient way to store two pieces of information in the same place.

AbiWord can export in DocBook format, which is an XML vocabulary for describing technical (computer) documents — it’s what I use for this
<!-- This DocBook file was created by AbiWord. -->
<!-- AbiWord is a free, Open Source word processor. -->
<!-- You may obtain more information about AbiWord at www.abisource.com -->
<!-- This is a DocBook 4.2 DTD file -->
<title>Compatibility</title>
<section role="unnumbered">
  <para>You can of course buy and install a fully-fledged commercial XML editor with XSLT support, and run this application within it. However, this is beyond the reach of many users, so to do this unaided you just need to install three pieces of software: <ulink url="http://java.com/download/"><emphasis>Java</emphasis></ulink>, <ulink url="http://saxon.sourceforge.net"><emphasis>Saxon</emphasis></ulink>, and the <ulink url="http://www.docbook.org/xml/4.2/index.html">DocBook 4.2 DTD</ulink> (URIs are correct at the time of writing). None of these has a visual interface: they are run from the command-line in the same way as is possible with
The XSLT language lets us create templates for each type of element in an XML document. In our example, there are only three which need handling, as we did not create chapter or section titles (DocBook requires them to be present, but they don’t have to be used).

- **para**, for the paragraph[s];
- **ulink**, for the URIs;
- **emphasis**, for the italicisation.

I’m going to cheat over the superscripting and subscripting of the letters in the L\(\text{A}\)\TeX\Xlogo, and use my editor to replace the whole thing with the \LaTeX\ command. In the other three cases, we already know how \LaTeX\ deals with these, so we can write the templates accordingly.

Writing XSLT is not hard, but requires a little learning. The output method here is **text**, which is L\(\text{A}\)\TeX\’s file format (XSLT can also output HTML and other flavours of XML).

```xml
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="2.0">
  <xsl:output method="text"/>
  <xsl:template match="/">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="book">
    <xsl:text>
\begin{document}
</xsl:text>
    <xsl:apply-templates/>
    <xsl:text>
end(document)
</xsl:text>
  </xsl:template>
  <xsl:template match="para">
    <xsl:apply-templates/>
    <xsl:text>&#xa;</xsl:text>
  </xsl:template>
  <xsl:template match="ulink">
    <xsl:text>
\footnote{url}</xsl:text>
    <xsl:value-of select="@url"/>
    <xsl:text>}</xsl:text>
  </xsl:template>
  <xsl:template match="emphasis">
    Formatting Information
  </xsl:template>
</xsl:stylesheet>
```
CHAPTER 8. COMPATIBILITY

1. The first template matches /, which is the document root (before the book start-tag). At this stage we output the text which will start the \LaTeX document, \texttt{\documentclass{article}} and \texttt{\usepackage{url}}.

The \texttt{apply-templates} instructions tells the processor to carry on processing, looking for more matches. XML comments get ignored, and any elements which don't match a template simply have their contents passed through until the next match occurs, or until plain text is encountered (and output).\footnote{Strictly speaking it isn't output at this stage: XML processors build a 'tree' (a hierarchy) of elements in memory, and they only get 'serialised' at the end of processing, into a stream of characters written to a file.}

2. The \texttt{book} template outputs the \texttt{\begin{document}} command, invokes \texttt{apply-templates} to make it carry on processing the contents of the \texttt{book} element, and then at the end, outputs the \texttt{\end{document}} command.

3. The \texttt{para} template just outputs its content, but follows it with a linebreak, using the hexadecimal character code \texttt{x0A} (see the ASCII chart in Table E.1 on page 269).

4. The \texttt{ulink} template outputs its content but follows it with a footnote using the \texttt{\url} command to output the value of the \texttt{url} attribute.

5. The \texttt{emphasis} template surrounds its content with \texttt{\emph{ and }}.

If you run this through \textit{Saxon}, which is an XSLT processor, you can output a \LaTeX file which you can \texttt{pdflatex}. For example:

\begin{quote}
$\texttt{java -jar saxon9.jar -o para.ltx para.dbk para.xsl}$
$\texttt{pdflatex para.ltx}$
\end{quote}

This is pdftex, Version 3.1415926-1.40.10 (\TeX Live 2009/Debian)
This is a relatively trivial example, but it serves to show that it's not hard to output \LaTeX from XML. In fact there is a set of templates already written to produce \LaTeX from a DocBook file at \url{www.dpawson.co.uk/docbook/tools.html#d4e2905}.
You can of course buy and install a fully-fledged commercial XML editor with XSLT support, and run this application within it. However, this is beyond the reach of many users, so to do this unaided you just need to install three pieces of software: 
\emph{Java} \footnote{\url{http://java.sun.com/j2se/1.4.2/download.html}}, \emph{Saxon} \footnote{\url{http://saxon.sourceforge.net}}, and the DocBook 4.2 DTD \footnote{\url{http://www.docbook.org/xml/4.2/index.html}} (links are correct at the time of writing). None of these has a graphical interface: they are run from the command-line in the same way as is possible with \LaTeX.

\end{document}

\section{Converting out of \LaTeX}

Before looking at one-way systems, see the earlier note about \textit{Pandoc} on page 175.

Converting \LaTeX to other formats is much harder to do comprehensively. As noted before, the \LaTeX file format really requires the \LaTeX program itself in order to process all the packages and macros, because
there is no telling what complexities authors have added themselves (what a lot of this book is about!).

Many authors and editors rely on custom-designed or homebrew converters, often written in the standard shell scripting languages (Unix shells, Perl, Python, Tcl, etc). Although some of the packages presented here are also written in the same languages, they have some advantages and restrictions compared with private conversions:

- Conversion done with the standard utilities (eg awk, tr, sed, grep, detex, etc) can be faster for *ad hoc* translations, but it is easier to obtain consistency and a more sophisticated final product using \texttt{lwarp, \LaTeX\2HTML} or \texttt{TeX4ht} — see below — or one of the other systems available.

- Embedding additional non-standard control sequences in \LaTeX\ source code may make it harder to edit and maintain, and will definitely make it harder to port to another system.

- All the above methods (and others) provide a fast and reasonably reliable way to get documents authored in \LaTeX\ onto the Web in an acceptable — if not optimal — format.

\LaTeX\2HTML was written to solve the problem of getting \LaTeX-with-mathematics onto the Web, in the days before MathML and math-capable browsers. \TeX\4ht was written to turn \LaTeX\ documents into Web hypertext — mathematics or not. The \texttt{lwarp} project aims to allow a rich \LaTeX\ document to be converted to a reasonable HTML5 interpretation, with only minor intervention on the user’s part.

There is a very useful list of all the alternatives in the \texttt{lwarp} package documentation (Dunne, 2018, p. 55–57)

### 8.2.1 Conversion to Word

There are several programs on CTAN to do \LaTeX-to-Word and similar conversions, but they do not all handle everything \LaTeX\ can throw at them, and some only handle a subset of the built-in commands of default \LaTeX\. Two in particular, however, have a good reputation, although I haven’t used either of them (I tend to stay as far away from \texttt{Word} as possible):
\begin{latex}

\textbf{CHAPTER 8. COMPATIBILITY}

\section*{\textbf{8.2.2 The \texttt{lwp} package}}

This \LaTeX package produces HTML5 output directly, using external utility programs only for the final conversion of text and images. Mathematics may be represented by SVG files or MathJax.

The \texttt{lwp} package is under active development and supports a wide range of formatting packages, but no attempt has been made to force \LaTeX to provide for every HTML-related possibility, as HTML cannot exactly render every possible \LaTeX concept.

\section*{\textbf{8.2.3 \texttt{\texttt{\LaTeXtoHTML}}} }

As its name suggests, \texttt{\texttt{\LaTeXtoHTML}} is a system to convert \LaTeX structured documents to HTML. Its main task is to reproduce the document structure as a set of interconnected HTML files. Despite using Perl, \texttt{\texttt{\LaTeXtoHTML}} relies very heavily on standard Unix facilities like the \texttt{NetPBM} graphics package and the pipe syntax. Microsoft Windows is not well suited to this kind of composite processing, although all the required facilities are available for download in various forms and should in theory allow the package to run — but reports of problems are common.

\end{latex}
8.2. CONVERTING OUT OF $\LaTeX$}

- The sectional structure is preserved, and navigational links are generated for the standard Next, Previous, and Up directions.

- Links are also used for the cross-references, citations, footnotes, ToC, and lists of figures and tables.

- Conversion is direct for common elements like lists, quotes, paragraph-breaks, type-styles, etc, where there is an obvious HTML equivalent.

- Heavily formatted objects such as math and diagrams are converted to images.

- There is no support for homebrew macros.

There is, however, support for arbitrary hypertext links, symbolic cross-references between ‘evolving remote documents’, conditional text, and the inclusion of raw HTML. These are extensions to $\LaTeX$, implemented as new commands and environments.

$\LaTeX2HTML$ outputs a directory named after the input filename, and all the output files are put in that directory, so the output is self-contained and can be uploaded to a server as it stands.

8.2.4 $\TeX$4ht

$\TeX$4ht operates differently from $\LaTeX2HTML$: it uses the $\TeX$ program to process the file, and handles conversion in a set of postprocessors for the common $\LaTeX$ packages. It can also output to XML, including Text Encoding Initiative (TEI) and DocBook, and the OpenOffice and WordXML formats, and it can create $\TeX$info format manuals.

By default, documents retain the single-file structure implied by the original, but there is again a set of additional configuration directives to make use of the features of hypertext and navigation, and to split files for ease of use. This is a most powerful system, and probably the most flexible way to do the job.

8.2.5 Extraction from PostScript and PDF

If you have the full version of Adobe Acrobat Reader (or one of several other commercial PDF products), you can open a PDF file created by $PDF\LaTeX$, select and copy all the text, and paste it into Word.
and some other wordprocessors, and retain some common formatting of headings, paragraphs, and lists. Both solutions still require the wordprocessor text to be edited into shape, but they preserve enough of the formatting to make it worthwhile for short documents. Otherwise, use the \texttt{pdftotext} program to extract everything from the PDF file as plain (paragraph-formatted) text.

### 8.2.6 Last resort: strip the markup

At worst, the \texttt{detex} program on CTAN will strip a \LaTeX\ file of all markup and leave just the raw unformatted text, which can then be re-edited. There are also programs to extract the raw text from DVI and PostScript (ps) files.

### 8.3 Going beyond \LaTeX\n
The reader will have deduced by now that while \LaTeX\ is possibly the best programmable typesetting system around, the \LaTeX\ file format is not generally usable with anything except the \LaTeX\ program. \LaTeX\ was originally written in the mid-1980s, about the same time as the Standard Generalized Markup Language (SGML), but the two projects were not connected. However, \TeX\ and \LaTeX\ have proved such useful tools for formatting SGML and more recently XML that many users chose this route for their output, using conversions written in the languages already mentioned in section 8.2 on page 182.

Unfortunately, when the rise of the Web in the early 1990s popularised SGML using the HTML, browser writers deliberately chose to encourage authors to ignore the rules of SGML. Robust auto-converted formatting therefore became almost impossible except via the browsers’ low-quality print routines.

It was not until 1995–7, when the XML was devised, that it again became possible to provide the structural and descriptive power of SGML but without the complex and rarely-used options which had made standard SGML so difficult to program for.

XML is now becoming the principal system of markup. Because it is based on the international standard (SGML), it is not proprietary, so it has been implemented on most platforms, and there is lots of free software supporting it as well as many commercial packages. Like SGML,
it is actually a meta-language to let you define your own markup, so it is much more flexible than HTML. Implementations of the companion Extensible Stylesheet Language (XSL) provide a direct route to PDF but at the expense of reinventing most of the wheels which \LaTeX\ already possesses, so the sibling XSLT can be used instead to translate to \LaTeX\ source code, as shown in the example in section 8.1.1 on page 177. This is usually much faster than writing your own formatting from scratch in XSL, and it means that you can take full advantage of the packages and sophistication of \LaTeX. A similar system is used for the Linux Documentation Project, which uses SGML transformed by the Document Style Semantics and Specification Language (DSSSL) to \LaTeX

The source code of this book, available online at www.ctan.org/tex-archive/info/beginlatex/src/ includes XSLT which does exactly this.
Installation

\texttt{TEX Live} installs very easily on all modern desktop, laptop, and server platforms. You can get a copy of the DVD from your local \TeX\ user group, or you can download the installation from \texttt{CTAN} in \url{www.ctan.org/tex-archive/systems/}

This course is based on using one of the following distributions of \TeX\ from the \TeX\ Collection DVD:

\textbf{\TeX\ Live}: for Unix & GNU/Linux, including Apple Macintosh OS X, and Microsoft Windows (this supersedes Thomas Esser’s \texttt{te\TeX} distribution which is no longer supported);

\textbf{Pro\TeX\t}: for Microsoft Windows (by Thomas Feuerstack) which includes Christian Schenk’s \texttt{MiK\TeX}, the \texttt{\TeX\Studio} editor, \texttt{Ghostscript}, and \texttt{GSview};

\textbf{Mac\TeX\}: for Apple Macs running OS X. \texttt{Mac\TeX} includes the \texttt{\TeX\shop} editor.

Other implementations of \TeX\ can be downloaded from \texttt{CTAN}, but the above are the standard systems. \TeX\ is included with all distributions of \TeX\.

The \TeX\ Collection DVD is issued annually by \texttt{TUG} in conjunction with many of the local \TeX\ user groups around the world (see \url{www.tug.org/lugs.html} for addresses), and edited by Sebastian Rahtz, Karl Berry, Manfred Lotz, and the authors of the software.
mentioned above. These people give an enormous amount of their personal time and energy to building and distributing these systems, and they deserve the thanks and support of the user community for all they do.

There is also a selection of commercial distributions you can buy, as described in Table 1 on page xxix: they all process \TeX\ identically, but there are some differences in size, speed, packaging, installation, support, and extra software provided.

If you can’t or won’t install \TeX\ (for example, your computer is corporate issue and locked down to prevent additional software being installed), there are several interactive online systems available such as Share\TeX\, \LaTeX\ Base, Overleaf, Papeeris, and others. These systems have an edit window and a PDF window, just like an installed editor, and all run \TeX\ exactly as if it was installed on your desktop.

One final thing before we start: always check to see if there is a more recent version of the installation program online. See the list item ‘Use the latest versions’ section A.4 on page 225 for more details.

A.1 Installing the software

A.1.1 Unix and GNU/Linux

Users of modern GNU/Linux systems don’t even need the \TeX\ Collection DVD, as \TeX\ installation packages are available online, built into the package manager for your system.

If your system has a graphical package manager (eg Ubuntu’s Synaptic or Software Center, see Figure A.1 on page 191), run it and install texlive-full, ghostscript, gv, okular, biber, kile (or texstudio or emacs), and jabref. Some of them may already be installed on your system. Go and have a cup of coffee while they automatically install all the necessary components.

The new Ubuntu Software Centre only allows one package to be installed at a time, so you may prefer to use a typed command in a console window (other Linux users may prefer this approach anyway). The command name varies from distribution to distribution; but two common ones are apt and yum:

```
sudo apt install texlive-full biber ghostscript gv
```
A.1. INSTALLING THE SOFTWARE

Figure A.1: Installing TeX Live from Synaptic, an Ubuntu package manager

Unfortunately, the Okular and Evince PDF/DVI viewers have cumbersome interfaces: Okular prematurely replaced two older and much better-designed viewers, kdvi and kpdf; but you could install qpdfview instead, which is faster and lighter.

After installation, run texconfig (see Figure A.2 on page 193) in a terminal window to adjust your local settings. This is a console utility, so type $texconfig just to adjust your own personal settings, or $sudo texconfig to adjust them system-wide (for all users). In the utility, use the arrow keys to go up and down the options, and the TAB key to jump to (and switch between) the OK, Cancel, and other ‘buttons’ at the foot of the screen. The spacebar or the Enter key selects a menu item or button. Most settings are correct as installed, but you might want to change one of the following:

- the first option, DEST, lets you specify whether you normally want to print straight onto the printer, or ‘print’ into a file (to attach to email or upload somewhere);

- the default paper size (the PAPER option), if the installed size is not your most common one (A4 or Letter);
Unix and GNU/Linux installers

I strongly recommend this method for all GNU/Linux users. You should only install from the TeX Collection DVD if you are using an older, hand-built, or commercial Unix system which has no package manager, or where the version provided by the package manager is seriously out of date. Installing for a multi-user system from the TeX Collection DVD for Unix requires root privileges and a good understanding of Unix systems management, and is beyond the scope of this book.

The date of the version can be a drawback: the Linux repository versions of TeX can be up to a year out of date, because of the enormous volunteer effort required to put them together. For new users this should not be a concern, as most updates do not seriously affect core facilities. If there are very recently-updated packages you badly need, you can install them separately, using the instructions in section 3.2 on page 59.

If you do decide to switch to the TeX Live DVD, make sure you completely uninstall and purge the texlive-full package first, otherwise your system will get hopelessly confused. This will leave your system in a conflicted state, because it will try to uninstall all the packages above that depend on TeX. The solution is a ‘shim’ Debian package that pretends that the Debian distribution of TeX Live is still installed, while actually using the DVD-installed version.

- the printer resolution (the MODE option), where you can adjust your printer settings; this allows you to fine-tune it for, say, a typesetter that you want to send output to instead of your own printer;

- in the DVIPS option you can adjust your printer OFFSET (left and top margins), which is useful for older, less accurate printers.

You may also need the REHASH option later on. It is used to update TeX’s fast-find database (see step 4 on page 64) after adding new or updated packages.

If your printer is a conventional home or office ink-jet or laser printer, and is not shown, the LaserJet5 setting (600dpi) is probably a good bet.
A.1 INSTALLING THE SOFTWARE

Figure A.2: Running the post-installation program texconfig

While still in the utility, you can test the margin settings in another window by running the testpage.tex document through \TeX\ (by typing $latex testpage$ and responding to the questions about paper size and double-sided printing). Print the resulting .dvi file with the command $dvips -f testpage | lpr$ and adjust the margins in texconfig if necessary. These adjustments are not needed with PDF output.

A.1.2 Apple Mac OS X

Double-click the MacTeX-yyyy-DVD.mpkg package in the mactex folder of the \TeX{} Collection DVD (replace the yyyy with the year of your distribution as shown on the DVD sleeve). Install the package in the normal Mac way by dragging the package icon onto the hard disk icon.

If you don’t have the DVD, download the MacTeX.pkg package from tug.org/mactex/mactex-download.html and click it in your Downloads to start the installation.

Depending on how your Mac is set up, you may be asked to download and install the developert tools during the \TeX{} installation. Do so.
When it’s all finished, open your Applications folder in the Finder and go to the \TeX subfolder and drag \TeXshop out onto your Dock. \TeXshop is the editor supplied with Mac\TeX which you use for writing your documents.

If you are going to use Adobe Acrobat Reader instead of Preview, make sure you clear the font cache and set the resolution to the system default (112dpi) otherwise you may get very weird displays.

If you are using the El Capitan version of the operating system, it is possible that \TeXshop will not recognise the folder where Mac\TeX installed the system. If you get an error claiming pdflatex does not exist in /usr/texbin, then use the \TeXshop Preferences > Engine menu to change the first Path Setting to /Library/TeX/texbin.

If you are unused to installing Mac software, here’s a click-by-click guide:

A.1.2.1 Download the Mac\TeX installer

1. In the \TeX Users Group web page for Mac\TeX at tug.org/mactex, click on Mac\TeX Download

2. Click on the MacTeX.pkg link to download the installer
A.1. INSTALLING THE SOFTWARE

3. The installer will be downloaded into your Downloads area

A.1.2.2 Running the installer

4. Click on the Downloads icon to see the package and click on it to start installation

5. The Introduction screen explains what to do

6. The ReadMe screen lists what will be installed
7. The License screen explains your rights: read it

8. Click Agree to continue

9. The Destination screen lets you choose where to install. Normally you want to install for all users, even if you are the only user on the machine
A.1. INSTALLING THE SOFTWARE

10. Pick the Standard Installation

11. Wait while the files are installed

12. Install the developer tools (this step will not occur if you have them already installed)
APPENDIX A. INSTALLATION

13. Agree to the developer tools license

14. The installer will check the location of the developer tools

15. The developer software will be installed
A.1 INSTALLING THE SOFTWARE

16. The Mac\TeX\ installation scripts will be run

17. The developer software installation will continue

18. The Mac\TeX\ installer will complete
Appendix A. Installation

19. You can now dispose of the installer package.

A.1.2.3 Move the TeXShop editor into the Dock

20. Open the Finder and locate the TeX folder in Applications. In it is the TeXShop application.

21. Drag and drop the TeXShop application into your Dock.

A.1.2.4 Testing MacTeX

\begin{document}
Hello, World!
\end{document}
22. Click \TeXShop in the Dock to open it. Type the 4-line document as shown above.

23. Set the \TeX processor to Xe\TeX in the drop-down menu and click \texttt{Typeset}. You will be asked to name and save the document: type a name, check the Encoding is set to UTF-8, and click Save.

24. The processing log will record what \TeX does (this is also where any error messages get shown).
APPENDIX A. INSTALLATION

Figure A.3: The Pro\TeX\t setup program on the \TeX\ Collection DVD

25. The typeset document is shown in a Preview window

A.1.3 Microsoft Windows

You can install Pro\TeX\t from the \TeX\ Collection DVD or from the downloadable installation from the \TeX\ Users Group web site.

A.1.3.1 Installing from the DVD with Autorun turned on

If your system has auto-run enabled, inserting the \TeX\ Collection DVD should start the setup program automatically. Click on the Open pro\TeX\t button to start. Continue from section A.1.3.5 on page 206 below.

If your system has auto-run turned off, insert the DVD, go to the protext folder, and double-click on the Setup.exe program as shown in Figure A.3 on page 202. Continue from section A.1.3.5 on page 206 below.
A.1. INSTALLING THE SOFTWARE

A.1.3.2 Downloading the Pro\TeX\xt installation program

Skip this if you have the Te\La\TeX\ collection DVD and go straight to section A.1.3.5 on page 206

1. In the Te\La\TeX\ Users Group web page for pro\TeX\xt at tug.org/protext/, click on download the self-extracting protext.exe file

2. Click on the protext.exe link to download the installer

3. The installer will be downloaded into your Downloads area
APPENDIX A. INSTALLATION

4. When it has finished downloading, wait while it is checked for malware. When the Run button appears, click it to start installation. You may be challenged by Windows Security asking if it is OK to run this program. Answer Yes.

A.1.3.3 Extracting the installation files

5. The first thing is to extract the installation files. Click on Browse to pick a folder.
A1. Installing the Software

6. Choose your Downloads folder and click OK

7. Click [Extract] to start extracting the installation files

8. The installation files will be unpacked to the folder you chose
APPENDIX A. INSTALLATION

A.1.3.4 Locating your unpacked installation files

9. When unpacking has finished, run your File Explorer. In older versions of Windows, this was called My Computer or just Computer.

10. Go to the folder where you unpacked the installation files and double-click on Setup.exe.

A.1.3.5 Installing pro\TeX t

By now you should be able to double-click the Setup.exe program in the pro\TeX t folder.

11. In the Pro\TeX t setup window, select your language and click on the MiKTeX Install button.
A1. INSTALLING THE SOFTWARE

12. Click in the box to accept the MiKTeX licence and click [Next] to continue.

13. Make sure the Complete MiKTeX option is selected and click [Next] to continue.

14. Choose a private installation or one that everyone who uses your computer can use, and click [Next] to continue.
APPENDIX A. INSTALLATION

15. Accept the installation folder that MikTeX suggests (unless you are an expert or have a special disk setup) and click [Next] to continue.

16. In the Options screen, select your paper size (A4 or US Letter), and whether or not you want extra packages to be downloaded and installed automatically (Yes or No) — on a laptop where a network connection is not always present, choose ‘Ask first’ instead, then click [Next] to continue.
17. Finally, accept the settings as shown (or change them by clicking [Back]), and then click [Start] to start the installation process.

![MiKTeX Installer](image1)

18. During installation, MiKTeX will list the files it is installing and show a progress bar.

![MiKTeX Files](image2)

19. When it is all done, wait while it indexes your fonts, then click [Finish].

![MiKTeX Font Index](image3)
20. Click [Close] to finish the installation of MiKTeX

![Completing the MiKTeX Setup Wizard](image)

### A.1.3.6 Installing \TeXStudio

21. Go back to the Pro\TeXt setup window and click on the \TeXStudio **Install** button

![Select Setup Language](image)

22. The \TeXStudio installation program will start and ask you to select the language to use during installation and click **OK** to continue
23. Click \texttt{Next} in the following screen to continue.

24. Accept the installation folder that \texttt{TeXStudio} suggests (unless you are an expert or have a special disk setup) and click \texttt{Next} to continue.

25. Also accept the suggested location for the \texttt{TeXStudio} shortcut and click \texttt{Next} to continue.
26. Add icons to your desktop and Quick Launch bar by selecting the boxes, then click [Next] to continue.

27. Finally, accept the settings as shown (or change them by clicking [Back]), and then click [Install] to start the installation process.

28. Wait a few moments while "TeXStudio" installs.
A.1. INSTALLING THE SOFTWARE

29. When it is all done, click Finish and TeXStudio will start.

30. Close down the installer.

A.1.3.7 Verifying the version of TeXStudio

31. If there is a new version available, you will be notified. You download it from the TeXStudio web site at texstudio.org/ as described below.
APPENDIX A. INSTALLATION

32. If \texttt{TeXStudio} claims it cannot find the installation of \LaTeX, it is out of date and should be replaced by an updated version from their web site as described here.

33. Download a new version from the \texttt{TeXStudio} web site at \url{texstudio.org}. Click on \texttt{Download now} to get the latest version.

34. Downloading a new version of \texttt{TeXStudio}
A.1. INSTALLING THE SOFTWARE

35. When the download has finished, click Run.

Continue the procedure from step 22 on page 210.

A.1.3.8 Testing TeXStudio

\documentclass{article}
\begin{document}
Hello, World!
\end{document}

36. When TeXStudio runs, click File \textgreater{} New and type the 4–line document as shown above.

37. Click the Options Configure TeXStudio menu and set the processor (Default Compiler) to \LaTeX.
APPENDIX A. INSTALLATION

38. Typeset the document by clicking on the green double-arrow icon (Build & View)

39. Shortcuts make life simpler

One last thing to do: add a personal TeX folder for extra downloads such as additional fonts.
A.2 Your personal \TeX\ directory

MiKTeX (Windows) can recognise when you try to use a class or package in your document that isn’t installed, and automatically download and install it for you right there and then, and carry on compiling your document. \TeX\ Live and Mac\TeX\ have a separate package-manager which can be used to add classes and packages if you find you need them.

Doing this is fairly rare if you picked the ‘full’ installation option, as you then have pretty much everything already installed, but there are always new packages coming out, and others being updated.

However, there are also times when you may want to add a new or uncommon class or package by hand — perhaps a private one from a company or organisation (so \LaTeX\ and CTAN wouldn’t know about it), or even one you are writing yourself.

To do this, you need a place to put the files where they won’t get mixed up with your documents or with \TeX\’s own files. This is the ‘right place to put files’ mentioned in step 3 on page 62, and it’s known as your personal \TeX\ directory or personal \TeX\ folder.

\LaTeX\ will automatically check this place first for classes and packages, so anything you put in your personal \TeX\ directory will be found before any file of the same name in your main \TeX\ installation, which is why it’s important for manual updates, and for special or private classes, packages, styles, and fonts.

The folder is called texmf (for \TeX\ and METAfont). For all Unix & GNU/Linux systems, including Apple Macintosh OS X, and for \TeX\ Live systems on Microsoft Windows, that’s all you have to create for now.

For MiKTeX (Pro\TeX\t, you also have to tell the system that you want the folder to be searched — see section A.2.3 on page 218: this is really important, as otherwise MikTeX won’t use it.

### A.2.1 Create a personal \TeX\ folder in Unix and GNU/Linux

Either open a terminal window and type $mkdir ~/texmf$

Or if you prefer to use a file-manager:

1. Open a file-manager window (eg Thunar, Nautilus, Dolphin, etc) on your Home directory;
APPENDIX A. INSTALLATION

2. Right-click in an empty area of your Home directory so the menu
dialog appears;
3. Click [Create] New or [New];
4. Click [Folder] or [Directory];
5. Name the new folder `texmf`;

A.2.2 Create a personal TeX folder in Apple Mac OS X

Either open a Terminal window (find `Terminal` in `Applications` > `Utilities`) and type `$mkdir ~/Library/texmf` 
Or if you prefer to use the Finder:
1. Open the Finder on your Home folder;
2. Click on `View` > `As Columns`;
3. Click on `View` > `Show View Options`;
4. In the Options dialog which appears, make sure `Show Library Folder` is checked, then close the dialog window;
5. Select `Library` in the list of folders;
6. Click `File` > `New Folder`;
7. Name the new folder `texmf`;
8. Close the Finder.

A.2.3 Create a personal TeX folder in Microsoft Windows

Either open a Command window (find `Command` in `All Programs`) and type

```
cd %HOME%
md texmf
```

Or if you prefer to use the graphical directory manager:
1. Open *My Computer* (just called *Computer* in *Windows 7/8*);

2. Create a new subfolder called *texmf* in the *C*: drive (Win95—XP) or *Computer\System\Users\your name\texmf* (Win7–10), as in Figure A.4 on page 219.

You now have to add this folder to the MiKTeX list of root folders:

3. Name the new folder *texmf* (make sure it is called *texmf* (all lowercase) and nothing else);

4. Click the *Start* or *Windows* button and run the MiKTeX Options (maintenance) program (it should be shown among your recent programs).

5. Click the *Roots* tab and the *Add* button, and navigate in the window to the place where you created the *texmf* folder above (Figure A.5 on page 220).

Finally, get MiKTeX to update it:
APPENDIX A. INSTALLATION

Figure A.5: Adding your Personal $\TeX$ Directory to MiKTeX

6. Click on the General tab and tell MiKTeX to update it along with its other folders by clicking the Refresh FNDB button (Figure A.6 on page 221).

You MUST click on the Refresh FNDB button any time you make changes to the contents of your personal $\TeX$ directory (the texmf folder), otherwise MiKTeX will not be able to find the files.
Figure A.6: Updating MiKTeX's FileName DataBase (FNDB)
APPENDIX A. INSTALLATION

A.3 Picking an Editor

One of the best features of \TeX-based systems like \LaTeX{} is that they don’t force you to use any particular editor or viewer: you can pick one that you’re comfortable with.

One of the worst features (for a beginner) is not understanding this: many new users have never come across this flexibility in software before, and may be unfamiliar with the idea that you don’t have to do what your vendor says: you can pick and choose.

Nevertheless, I’m solving this by edict for beginners here: unless you already have a pet editor or viewer, just use the one shown below. If you’re an experienced computer user, see the comments at the end of this section.

So for beginners, in the case of Pro\TeX{}t (Windows) and Mac\TeX{} (Macs) the editor is the one that comes with the distribution (\TeX{}Studio and \TeXshop{} respectively). Unix and GNU/Linux users need to choose and install an editor separately.

<table>
<thead>
<tr>
<th>System</th>
<th>Package</th>
<th>Engine</th>
<th>Editor</th>
<th>Viewer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Windows</td>
<td>Pro\TeX{}t</td>
<td>\MiK\TeX{}</td>
<td>\TeX{}Studio</td>
<td>Adobe Acrobat Reader</td>
</tr>
<tr>
<td>Apple Mac OS X</td>
<td>Mac\TeX{}</td>
<td>\gaw\TeX{}</td>
<td>\TeXshop</td>
<td>Preview</td>
</tr>
<tr>
<td>Unix and GNU/Linux</td>
<td>\TeX{} Live</td>
<td>\TeX{} Live</td>
<td>Kile, \TeX{}Studio, or \Emacs</td>
<td>Okular, Evince, or qpdfview</td>
</tr>
</tbody>
</table>

However, there are many other editors, if you want to try them out and pick one you are comfortable with. In particular, \Emacs is also available for Windows (and for Mac OS X — Aquamacs); \TeX{}Maker is also available for Macs; and the \I\TeX{} near-WYSIWYG front-end to \I\TeX{} is available for all platforms.

Have a look at Barbara Beeton’s slide on Features of a good editor, slide 4 from her presentation at TUG 2017 (Beeton, 2017).

‘Choose your editor carefully’ Experienced computer users should also read Michael Sperberg-McQueen’s very sensible and relevant comments: Twenty years ago, when my computer center required
MiKTeX and TeXnicCenter

If you plan on using the TeXnicCenter editor instead of TeXStudio, you must make sure when you install MiKTeX, that you make a careful note of the folder you install MiKTeX into because you will need that later, when you run TeXnicCenter for the first time after installation. It should be something like C:\Program Files\MiKTeX 2.9\miktex\bin (or a later version number, if this has moved on since the time or writing). You need to know this, because TeXnicCenter won’t guess it for you.

us all to start moving our data from VM/CMS to Unix, and we were all trying to decide which of the available text editors to invest time in learning, I formulated one of the few general principles of computer usage that has ever been revealed to me:

1. If you are a serious computer user, you will spend more time in your text editor than in any other single application. Choosing well is a good idea.

2. It takes a long time to learn an editor well. You will not learn 2000 editing programs between now and the day you stop using computers at all — you may only learn two or three. Choosing wisely is important to avoid wasting time.

3. The first question to ask, when considering any candidate editor, is ‘when I write a macro for this editor, what is the name of the language I’m writing in?’

There are three classes of answer:

☐ Answers of the form ‘you don’t, there are no macros in ———’. These won’t normally occur, since such editors won’t ever actually be candidates for serious day-in day-out use.

☐ Answers of the form ‘It’s a specialized macro lang-[…]’. You needn’t listen to the end of these; the editor is not worth investing your time to learn.

☐ Answers of the form ‘X’, where ‘X’ is the name of a ‘real’ programming language, with variables and functions and flow of control and all of those things.

It was on this basis that I chose Emacs over vi. (A decision, I may say, that I have never regretted.)
The importance of macros was pretty clear to me because over my years of using VM, I had written dozens of macros for XEdit, the system editor, and the fact that I could write them in a well designed programming language (Rexx) was really important.

The funny thing is that having chosen Emacs over vi because it has a real programming language for macros, I then used Emacs for ten years or more before I ever actually used Lisp to write any macros. I decided that this was because all the macros I needed seemed to have been written by someone else already.

### A.4 Installation problems

It’s always annoying when a program that’s supposed to install painlessly causes trouble, and none the more so when everyone else seems to have been able to install it without problems. I’ve installed \TeX\ hundreds of times and very rarely had any difficulties, but these are a few of the occasions when I did.

**Bad hard disks:** If you are using Microsoft Windows, you should run a scan and defragmentation of your hard disk[s] before you start. It should take under an hour on a modern machine unless you have a very large disk, but it may need overnight on an older machine. Clean your DVD drive if it has been in heavy use. \TeX\ is made up of a very large number of very small files, so there is a lot of disk activity during an installation. Microsoft Windows runs very slowly when installing a lot of small files, so be patient.

On any system, if you are installing a new hard disk for your typesetting work, you have the chance to reformat it beforehand. Pick the smallest granularity (cluster size) possible, usually 1024 bytes (1Kb). This minimises the space needed for systems with a very large number of very small files like \TeX\ has, and may help improve the speed and reliability of the system.

**Windows Registry errors:** This only affects Microsoft Windows users. The Registry is where Microsoft wants software companies to store details of all the programs you install. Unfortunately the Registry is grossly abused by marketing departments to try and foist undesirable links on you, the user. You will see this with...
A.4. INSTALLATION PROBLEMS

many commercial programs, where a particular type of file you’ve been able to double-click on for years suddenly runs a different program. Some programs install obsolete or broken copies of program libraries (DLL files), overwriting ones which were working perfectly. Worse, the viruses, trojans, and worms which typically infect unprotected Windows systems can leave unwanted links to web pages, or change some of the ways in which Windows operates. The overall effect can be that the whole machine slows down, or that files which are expected to do one thing do another. The best solution is a thorough Registry clean-out, using one of the many free or commercial programs available for the purpose.

Use the latest versions: Before installing, check the CTAN web site at www.ctan.org/ for the latest version of Pro\TeXt (Windows), Mac\TeX (Macs), or \TeX Live (all platforms) for the latest copy of the installation program. Just occasionally a bug slips through onto the production DVD, and although it’s always fixed and notified on comp.text.tex, that’s a high-volume newsgroup and even the sharpest eyes may miss an announcement.

Unix and GNU/Linux users will always get the latest repository copy from their system’s package manager, but this may not be the absolute latest copy of \TeX (see section A.1.1 on page 192 for why). If you are installing on Unix manually from the \TeX Collection DVD instead, check on CTAN for an updated version of the file install-tl.sh.

Stick to the defaults: Unless you’re a computer scientist or a software engineer, I very strongly suggest you never change or fiddle with the default directories for installation. I know some of them look odd, but they’re that way for a purpose, especially when it comes to avoiding folder names with spaces in them, like the notorious C:\Program Files. Although most modern systems cope happily with spaces in filenames and directory names when using a graphical user interface, they are always A Bad Idea, especially for programs which can be run from scripts (\TeX is one). Spaces and other non-alphanumeric characters should therefore be avoided like the plague (they are forbidden in web addresses [URIS] for the same very good reason: the people who designed
them knew the pitfalls). It may look snazzier to put the installation in My Cute Stuff, but please don’t: you’ll just make it harder to find, harder to fix problems, and more embarrassing if you have to explain it to someone else trying to help you.

64-bit Windows: The MiKTeX distribution for Windows is a 32-bit program but it should install correctly on 64-bit Windows 7 systems. For safety, close down all other programs before starting the installation.

Locked systems: If you want to install ProTeXt on a computer in a lab or other group environment where the disk storage is locked down, and where the Administrator is unwilling or unavailable to install it for you, there are a couple of choices:

- Install it on a USB stick that you can unplug and carry with you. That way your \LaTeX{} installation is always with you. If you use it on another computer where the USB device mounts as a different disk letter, you will need to configure it so that it can ‘see’ where it is in the directory system.
- If you cannot install it at all, because the Windows Registry is also locked, and the Administrator is unwilling or unable to install it for you, you may be able to install it in a virtual container (eg Windows XP as a virtual image inside Windows 7). It will be slow, and it may be missing some facilities like alternate character sets, but it will execute.

Bear in mind that shared systems in large companies, universities, and similar organisations do usually prohibit software being installed by the user (you) because of security issues over viruses, support, maintenance, and other factors. If you feel your institution needs a network installation of \LaTeX{}, ask your Administrator or IT Centre to contact the \TeX{} Users Group or any local use group (see Appendix 4 starting on page 261), who may be able to help.

A.5 Configuring \TeX{} search paths

\TeX{} systems run on a huge variety of platforms, and are typically made up of a very large number of very small files in several separate
A.5. CONFIGURING \TeX{} SEARCH PATHS

'trees' of directories (folders). This allows users to update parts of the system without having to update all of it, and to maintain their own tree of preferred files without having to have administrative rights on the computer.

To make sure \TeX{} finds the right file, it uses a technique borrowed from the Unix world, based on a simple 'hash index' for each directory tree they need to look in. This is known as the ls-R database, from the Unix command ($\texttt{ls -R}$) which creates it. The program which does this for \TeX{} is actually called after this command: \texttt{mktexlsr}, although it may be aliased as \texttt{texhash} or something else on your system. This is the program referred to in step 4 on page 64.

However, to know where to make these indexes, and thus where to search, \TeX{} needs to be told about them. You don’t normally need to change the configuration, but sometimes you might want to move directories between disks to free up space or use faster equipment, which would mean changing the configuration.

In a standard \TeX{} installation this information is in the main (not the local) installation directory, in \texttt{texmf/web2c/texmf.cnf}. The file is similar to a Unix shell script, but the only lines of significance for the search paths are the following (this is how they appear in the default Unix installation, omitting the comments):

```
TEXMFMAIN = /usr/share/texmf
TEXMFLOCAL = /usr/local/share/texmf
HOMETEXMF = $HOME/texmf
TEXMF = {$HOMETEXMF,!!$TEXMFLOCAL,!!$TEXMFMAIN}
SYSTEXMF = $TEXMF
VARTEXFONTS = /var/lib/texmf
TEXMFDBS = $TEXMF;$VARTEXFONTS
```

This defines where the main \TeX{}/METAFONT (\texttt{texmf}) directory is, where the local one is, and where the user’s personal (home) one is. It then defines the order in which they are searched, and makes this the system-wide list. A temporary directory for bitmap fonts is set up, and added to the list, defining the places in which \texttt{texhash} or \texttt{mktexlsr} creates its databases.

In some installations, the local directory may be set up in a slightly different directory to the one given in the example. Under Microsoft Windows, the names will be full paths such as
APPENDIX A. INSTALLATION

Finding out where \LaTeX looks for stuff

There is a program distributed in all \LaTeX installations called kpsewhich, a \TeX-specific variant of the standard Unix which (1) command. If you type the command followed by a filename, it will tell you whereabouts in your \TeX installation it is.

$ kpsewhich article.cls
/usr/share/texmf-texlive/tex/latex/base/article.cls$

Better, there is an option to tell you where your main and local trees (directories) are installed, and even where \LaTeX puts its map files and format files (internal setups):

$ kpsewhich --expand-var '$TEXMFMAIN'
$ kpsewhich --expand-var '$TEXMFLOCAL'
$ kpsewhich --expand-var '$TEXMFSYSVAR'$

So if you forget where to put something you are installing, `$TEXMFLOCAL` is where it should go.

C:\Program Files\TeXLive\texmf (for \TeX Live) or C:\Program Files\MikTeX 2.9\tex (for MiK\TeX). On an Apple Mac, it is `/Library/texmf' for each user.
Installing new fonts

Directories (Folders)

In the examples below, all the folders (directories) are assumed to be in your Personal TEX Directory unless otherwise explicitly given. See section A.2 on page 217 for how to find this out.

Different fonts come in a variety of packagings: the most commonly used with \TeX\ systems are \textit{PostScript} fonts and \texttt{METAFONT} fonts, but \TeX\ also lets you use TrueType and OpenType fonts. How you install them and where they go depends on how you installed \TeX: all I can deal with here are the standard locations within the \texttt{TDS}. These typefaces come supplied as one or more font ‘outline’ files and a number of ancillary files:

\texttt{METAFONT} typefaces: have a number of \texttt{.mf} source (outline) files, possibly also some \texttt{.fd} (font definition) files and a \texttt{.sty} (style) file. The \texttt{.tfm} (\TeX\ font metric) files are not needed at installation, as they get generated from the outlines automatically the first time you use the font.

\texttt{PostScript} typefaces: come as a pair of files: a \texttt{.pfb} (PostScript font \textit{binary}) or \texttt{.pfa} (PostScript font \texttt{ASCII}) outline, and an \texttt{.afm}
APPENDIX B. INSTALLING NEW FONTS

(Adobe font metric) file. There may also be .inf and other files but these are not needed for use with \TeX{} systems.

**TrueType and OpenType typefaces:** are a single .ttf or .otf file, which combines outlines and metrics in one.

The instructions for Type 1 and METAFONT typefaces here assume the use of the New Font Selection Scheme (NFSS) used in \LaTeX{} 2e. If you are running the obsolete \LaTeX{} 2.09, upgrade it now, because none of this will work.

As TrueType and OpenType are the easiest to deal with, I'll mention them first.

### B.1 TrueType and OpenType fonts

This is the simplest, as they are already recognised by your computer, so you just install them in the normal way for your computer system (usually double-click on the font file and select [Install] from the menus).

On Windows systems, the fonts are made available automatically; on UNIX & GNU/Linux systems, including Apple Macintosh OS X, run the *fc-cache* utility to give Xe\LaTeX{} fast-load access to all the fonts you have installed. Run the program like this:

```
fc-cache -fv
```

It can take several minutes, especially if you have a lot of fonts. Once it has finished, you can query the database with the $fc$-list command and $grep$ for the string you want:

```
fclist|grep -i comic|sort
/home/peter/texmf/fonts/type1/comicsans/rcomic8r.pfb: Comic Sans MS:
/home/peter/texmf/fonts/type1/comicsans/rcomicbd8r.pfb: Comic Sans MS:
/usr/share/texlive/texmf-dist/fonts/type1/rozynski/comicneue/ComicNeue...
```

### B.2 Installing METAFONT fonts

This is the simplest installation. When you download METAFONT fonts from CTAN, you'll usually find a number of outline files (.mf files) and maybe some other types as well (see below).
B.3 INSTALLING POSTSCRIPT FONTS

Installation of METAFONT fonts

1. In your Personal TEX directory, create a new subdirectory called `fonts/source/public/name` named after the typeface you're installing.

2. Copy all the `.mf` files to this directory;

3. Copy any `.fd` file[s] to your `tex/latex/mfnfss` subdirectory (create it if it doesn’t already exist);

4. Copy any `.sty` (style) files to a subdirectory (create it too), named after the typeface, eg `tex/latex/name`;

5. If you are using MiKTeX, run your Ti\TeX\ indexer program (see step 4 on page 64).

   UNIX & GNU/Linux systems, including Apple Macintosh OS X, don’t need this last step.

That’s it. Unlike PostScript fonts, METAFONT fonts generate their font metric files (.tfm files) automatically on-the-fly the first time the font is used, so there should be nothing else to install.

Now you can put a \usepackage command in your Preamble with whatever name the `.sty` file was called, and read the documentation to see what commands it gives to use the font (refer to section 3.2.1.2 on page 60 and step 2 on page 62).

If the font came without `.fd` or `.sty` files, you’ll need to find someone who can make them for you (or follow the outline in section B.3 on page 231, step 8 on page 238).

B.3 Installing PostScript fonts

Lots of people will tell you that PostScript fonts and PostScript output are dead and that TrueType or OpenType fonts and PDF output are the way to go. While this may be true for some cases, standard \TeX\ does

---

\footnote{On UNIX & GNU/Linux systems, including Apple Macintosh OS X, the easiest way to do this is in a Terminal window, in your Personal \TeX\ Directory, using the command `mkdir -p fonts/source/public/whatever`, as this creates any intervening subdirectories for you. Under Windows, you have to create each subdirectory individually.}
not work with TrueType fonts and does not produce PDF directly. Only \TeX\ and \LaTeX\ do that, and there are still many printers whose typesetters and platemakers require \TeX\ files rather than PDF. In addition, operating system support for scalable fonts is still very poor on Unix and GNU/Linux systems, despite the advances in recent years, and many rebranded (‘knock-off’ or pirated) TrueType or OpenType fonts supplied with other systems are of very poor quality. So in many cases it still makes sense to use \TeX\’s built-in support for PostScript fonts. When \TeX\ becomes the default processor (it’s currently still \LaTeX\), it will no longer be necessary to install different types of font files separately, as \TeX\ is able to use all fonts natively from your system’s font folders.

Two files are needed for each \TeX\ Type 1 font: the \texttt{.afm} Adobe Font Metric (AFM) and the \texttt{.pfb} PostScript Font Binary (PFB) files. You must have both types of file for each separate font before you start. If you only have the near-obsolete \texttt{.pfa} PostScript Font ASCII (PFA) files, it may be possible to generate the \texttt{.pfb} files using the \texttt{t1binary} program from the \texttt{tlutil} suite (see gnuwin32.sourceforge.net/packages/tlutil.htm) or the excellent \texttt{FontForge} font editor (from fontforge.sourceforge.net). There are unfortunately still some companies distributing Type 1 fonts in \texttt{.pfa} format (Mathematica is one reported recently).

I’ll repeat this: before you start, make sure you have all the \texttt{.afm} and \texttt{.pfb} files for the typeface you want. In the example below, I’m going to use a single font from an imaginary typeface called Foo, so I have \texttt{foo.afm} and \texttt{foo.pfb} files.

\textbf{Installation of PostScript Type 1 fonts: Preparation}

This part is very quick, but needs explaining.

1. **Put the files into a temporary directory**
   This is /\texttt{tmp} on Unix & GNU/Linux, including Apple Macintosh OS X; C:\texttt{\tmp} or C:\texttt{\temp} on Windows 95–XP; and should be Computer\textbackslash System\textbackslash Users\textbackslash your\_name\textbackslash \tmp on Windows 7 and up.

2. **Decide on the short font name to use inside \LaTeX.**
   This is not the full descriptive name (eg Baskerville Italic Bold Extended) but an encoded font name in the format \texttt{fnnsseev},
B.3. INSTALLING POSTSCRIPT FONTS

devised by Karl Berry, which stores the same information in no more than eight characters for compatibility with systems which cannot handle long filenames (and incidentally makes it far easier to type). The letters in the format above have the following meanings (see the fontname documentation on your computer for more details). Lists of the codes used are in the files supplier.map, weight.map, width.map, variant.map, and the various .map files for each foundry, which are in your \TeX{} installation.

<table>
<thead>
<tr>
<th>Letter</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>f</td>
<td>foundry</td>
<td>b=Bitstream, m=Monotype, p=Adobe</td>
</tr>
<tr>
<td>nn</td>
<td>typeface name</td>
<td>ba=Baskerville, tm=Times, pl=Palatino</td>
</tr>
<tr>
<td>ss</td>
<td>series/shape</td>
<td>r=roman, b=bold, i=italic, etc</td>
</tr>
<tr>
<td>ee</td>
<td>encoding</td>
<td>8r=\TeX{}Base1, 8t=Cork</td>
</tr>
<tr>
<td>v</td>
<td>variant</td>
<td>smallcaps, outline, script, etc</td>
</tr>
</tbody>
</table>

The fonts/map/fontname directory in your main (not local) installation directory of \TeX{} has files for several foundries giving fully-formed names like these for common fonts (eg ptmr8r is [Adobe] PostScript Times Roman in an 8–bit revised \TeX{} encoding; bgsslly is Bitstream Gill Sans Light in Y&Y’s \TeX{}n’ANSI encoding [LY1]).\footnote{Confusingly, Bitstream fonts (and others from similar sources) mostly have different names from the original fonts, to avoid copyright issues, so what they call Humanist 521 is actually Gill Sans. Until recently, US law only allowed the names of typefaces to be copyrighted, not the font designs themselves, leading to widespread piracy.} Read the documentation in Fontname: Filenames for \TeX{} fonts to find out how to make up your own short names if the foundry and font you want is not shown in the lists in the fonts/map/fontname directory.

In this example we’ll call our mythical example typeface ‘zork’ (standing for Zfonts Ordinary Bookface: z for a foundry otherwise undefined; or for Ordinary; and k is the letter used for Book designs, b being already the code for bold). We’ll assume the font comes in the two files foo.afm and foo.pfb that I mentioned above.

While the font/map/fontname directories have ready-made maps of these names for popular collections of typefaces, making
them up requires some knowledge of typographic terms and a careful reading of the *fontname* documentation.

3. **Decide on your encoding**

  Encoding is needed because Adobe fonts store their characters in different places to the \TeX{} standard. This is what tripped me up the first few times until someone pointed me at Y\&Y’s \TeX{}’n’ANSI encoding which at the time was the only one that included the glyphs I want where I expected them to be.\(^3\) Now, however, I recommend using the 8r encoding for all PostScript fonts. The encoding vector file 8r.enc should be in your main (*not* local) \TeX{} installation directory in `fonts/enc/dvips/base`.

  To avoid having to type the long path each time below, just copy this file to the temporary directory where you’re doing all this stuff.

Now you’re ready to convert the files to \TeX{} format.

**Installation of PostScript Type 1 fonts: Conversion**

1. **Convert the `.afm` files to `.tfm` and `.vf` format**

   The Adobe Font Metric files have to be converted to \TeX{} Font Metric and Virtual Font files. The `afm2tfm` and `vptovf` programs are standard \TeX{} utilities in the `bin` directory of your main \TeX{} installation.

   In a command window, we type:

   ```
   afm2tfm foo.afm -v zork8r.vpl -p 8r.enc rzork8r.tfm >zork.id
   ```

   This reads `foo.afm` using the 8r.enc encoding file, and creates a special ‘raw’ \TeX{} Font Metric file (hence the `r` prefix on `rzork8r.tfm`) that \TeX{} can use, with a list of all its properties encoded with 8r in the `.vpl` or Virtual Property List file.

---

\(^3\) Y\&Y, Inc has ceased trading and their \TeX{} distribution is not longer available, although there is email support at [lists.ucc.ie/lists/archives/yandytex.html](http://lists.ucc.ie/lists/archives/yandytex.html), and their encoding files continue to be used.

\(^4\) The only one I had problems with is ‘Å’, which for some weird reason isn’t catered for in this encoding.
Many people will tell you that virtual fonts are dead and that this is the wrong way to do it, but no-one has ever shown me an alternative that works, so I stick with it.

2. **Small caps (optional)**

   If you want a small caps variant faked up (perhaps because the typeface family doesn’t have a real small-caps font), repeat the medicine like this:

   \begin{verbatim}
   afm2tfm foo.afm -V zork8rc.vpl -p 8r.enc rzork8r.tfm >>zork.id
   \end{verbatim}

   Note the capital `V` option here. Yes, it *does* overwrite the `rzorkly.tfm` created in the first command — it doesn’t matter; let it (it will be the same information, anyway). And those are two ‘greater-than’ signs before the `zork.id` filename because we want to append to it, not overwrite it.

3. **Create the virtual font**

   The `vptovf` program turns the `.vpl` files into `.vf` and `.tfm` pairs. \LaTeX{} uses these to map the character positions from Adobe’s encoding to its own.

   \begin{verbatim}
   vptovf zork8r.vpl zork8r.vf zork8r.tfm
   vptovf zork8rc.vpl zork8rc.vf zork8rc.tfm
   \end{verbatim}

   Now we can install the files.

**Installation of PostScript Type 1 fonts: Installation**

1. **Make directories to hold the files**

   Under your Personal \TeX{} Directory, create a `fonts` subdirectory, and in it, create an `afm`, `tfm`, `typel`, and `vf` subdirectories (they may already exist from a previous font installation);

2. In each of these four, create a subdirectory for the foundry, making up a name if it’s not an established one (see the `suppliers.map` file);

3. Within each of the foundry directories, create a directory for the typeface (using an abbreviated human-readable typeface name,
APPENDIX B. INSTALLING NEW FONTS

Not the short Karl Berry fontname). On my computer, this means doing:

```
cd ~/texmf/fonts
mkdir -p afm/zfonts/zork
mkdir -p tfm/zfonts/zork
mkdir -p type1/zfonts/zork
mkdir -p vf/zfonts/zork
cd /tmp
```

Under Windows, where the $mkdir has no /p option, you probably have to create each subsubdirectory individually.

4. **Copy the files to their rightful places**

Copy the four groups of files to the four new directories:

```
cp *.afm ~/texmf/fonts/afm/zfonts/zork/
cp *.tfm ~/texmf/fonts/tfm/zfonts/zork/
cp *.pfb ~/texmf/fonts/type1/zfonts/zork/
cp *.vf ~/texmf/fonts/vf/zfonts/zork/
```

where ~/texmf is the root of your Personal TeX Directory. You can of course do all this with a directory window and mouse if you find it easier.

Last stage is to create the font map, style file, and font definition files.

**Installation of PostScript Type 1 fonts: Configuration**

1. **Create a font map**

   The font map is what tells TeX’s font driver which PFB file to use for which font.

   Open your editor and create a new file (if it asks for a name, call it zor.map);

2. Font entries MUST be on a single line each, with no line-breaking. Each entry has five fields separated by a space: a) the short fontname; b) the full (Adobe) font name; c) the PostScript encoding parameters (in quotes, with a leading and trailing space); d) the encoding file used; and e) the name of the PostScript outline
B.3. INSTALLING POSTSCRIPT FONTS

The TeXBase1Encoding ReEncodeFont can be used to convert a PostScript font to TeXBase1Encoding. The TeXBase1Encoding ReEncodeFont is a PostScript file. Fortunately, we captured almost all of this in those .id files in step 2 on page 235, so all you need to do is copy all those .id files into this file (eg $cat *.id >> zor.map), and add a space and the less-than sign and the name of the appropriate .pfb file to the end of each line, eg

\begin{verbatim}
 rzorkR ZorkOrdinary-Book "TeXBase1Encoding ReEncodeFont " <8r.enc <zork.pfb
\end{verbatim}

3. Save the file in fonts/map in your Personal \TeX\ Directory;

4. Update the map file into your \TeX\'s file maps with the command:

\begin{verbatim}
 updmap --enable Map=zor.map
\end{verbatim}

5. Create a style file

\TeX needs a style file to implement the interface to the font. Call it after the typeface or something related; in this example we\'ll call it foozork.sty and it needs to be saved into \texttt{tex/latex/zork/} in your Personal \TeX\ Directory.

In it go some details of the name and date we did this, what version of \TeX\ it needs, and any other command necessary to operate the font, like the font encoding and whether it is to supersede the current default Roman font.

Use your editor to open (create) this file;

6. Insert the following lines (obviously replacing the foozork stuff with meaningful values for your own fonts):

\begin{verbatim}
% foozork - LaTeX style file for the Zork font
\def\fileversion{1.0}
\def\filedate{2016/12/03}
\def\docdate{2016/12/03}
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{foozork}[\filedate\space fileversion\space
\ Zfonts Ordinary PSNFSS2e package]
\RequirePackage[T1]{fontenc}
\renewcommand{\rmdefault}{zor}
\endinput
\end{verbatim}
APPENDIX B. INSTALLING NEW FONTS

Note the following:

☐ The first argument to \ProvidesPackage MUST be the same as this style file name.

☐ If this is a typewriter font, change the renewed command \rmdefault into \ttdefault. If it’s a sans-serif font, make it \sfdefault instead. Omit this command completely if you don’t want the style file to supersede the current defaults but simply to make the font available for manual use.

☐ If you do that, you probably want to write a new command or two to use it, typically one unscoped command for grouped use and one scoped one for argument use:

\newcommand{\zork}{\fontencoding{T1}\%
\fontfamily{zor}\selectfont}
\newcommand{\textzork}[1]{{\zork#1}}

7. Save and close the file.

8. **Create the Font Definition file**

The last file needed is the font definition (.fd) file. This is named following the pattern eefnn.fd, using the same conventions as before, by prepending the (lowercase) encoding abbreviation to the foundry letter and fontname abbreviation, so our example would be t1zor.fd for the T1 (8r) encoding and the zor short fontname.

Use your editor to create the file in /texmf/tex/latex/zork (replacing zork with the name of the directory you created there).

9. Enter the following lines (see the notes below about what they mean):

\ProvidesFile{t1zor.fd}[2016/12/03 v0.1 manual font definitions for T1/zor.]
\DeclareFontFamily{T1}{zor}{}`
B.3. INSTALLING POSTSCRIPT FONTS

\begin{verbatim}
\DeclareFontShape{T1}{zor}{k}{n}{<-} {zork8r}
\DeclareFontShape{T1}{zor}{k}{sc}{<-} {zork8rc}
\end{verbatim}

10. Save and close the file.

Now you can \usepackage\{foozork\} in your \LaTeX\ file to make it the default font. To use the font incidentally instead of as the default, you can use the commands you added at the end of step 5 on page 237:

\begin{verbatim}
This is \{zorkfamily ZORK\} or \textzork\{ZORK\}
\end{verbatim}

B.3.1 Font Definitions

FD files typically use one \DeclareFontFamily command which specifies the encoding and the short font name. This is followed by as many pairs of \DeclareFontShape commands as you converted fonts (assuming you did both normal and small caps for each font: see step 2 on page 235; if you didn’t, then only one such command per font is needed here). The arguments to the \DeclareFontShape command to watch are the 3rd (weight/width, here k for Book, remember?), 4th (shape), and 5th (the size range and the name of the entry in the font map, minus the r prefix): the rest are static for each .fd file and simply identify the encoding and the font family.

The codes to use are given on pages 414–15 of the Companion and should also be in your copies of weight.map and width.map. The rules for combining weight and width need care: read the documentation for the fontname package. There is no shape.map in fontname because it’s not part of font file names, it’s purely a \LaTeX\ creation, so here’s what the same book says:

<table>
<thead>
<tr>
<th>Character</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>normal (upright)</td>
</tr>
<tr>
<td>it</td>
<td>italic</td>
</tr>
<tr>
<td>sl</td>
<td>slanted</td>
</tr>
<tr>
<td>sc</td>
<td>small caps</td>
</tr>
<tr>
<td>ui</td>
<td>upright italic</td>
</tr>
<tr>
<td>ol</td>
<td>outline</td>
</tr>
</tbody>
</table>

\textbf{Formatting Information}
APPENDIX B. INSTALLING NEW FONTS

Add your own for other oddities, but be consistent: I use \texttt{cu} for cursive (scripts), for example, and \texttt{k} for blackletter faces (not to be confused with \texttt{k} as a \textit{width} for ‘book’).

The default size range \texttt{<-->} in the fifth argument of the command \texttt{\DeclareFontShape} means that all sizes are to come from the same font outline (remember if this was a METAFONT font with different design sizes like CM it would be much more complex).

If the face has only a few variants, you can create any other entries for bold, italic, slanted, etc with the relevant weight and width and shape values pointing at the relevant outline file.

If you want one font to substitute for a missing one (for example italics to substitute for slanted in a typeface which has no slanted variant of its own) give the \texttt{ssub} (‘silent substitution’) command in the fontspec: for example to make all references to \texttt{sl} (slanted) type use an existing italic font, make the 5th argument like this:

\begin{verbatim}
{<--> ssub * zor/k/it}
\end{verbatim}

If you find the x-height of a font too big or too small to sort well with another font you are using, you can specify an \texttt{s} (‘scale’) factor in this argument instead: this example will shrink the result to 80\% of normal:

\begin{verbatim}
{<--> s * [0.8] zork8r}
\end{verbatim}

B.4 Updating your font maps

Because \LaTeX{} can handle so many different types of font file, it needs to know what to do when you use a particular typeface. We already saw in section B.1 on page 230 how to re-index your system’s font cache (Unix and GNU/Linux). That tells \LaTeX{} \textit{where} to find a font; the font map tells it \textit{what to do with it} when it’s found it.

Every font family or typeface prepared for use with \LaTeX{} \textbf{MUST} have a \texttt{.map} file to do this with, except METAFONT fonts, which don’t need them (we saw in step 4 on page 237 how to install the map file we created for a Postscript fonts).
B.4. Updating your font maps

B.4.1 Updating font maps on Mac and Linux (and \TeX Live on Windows)

In Unix and GNU/Linux systems (including Mac OS X) it’s very simple, as we saw in step 6 on page 159:

```
updatemap --enable Map=xxxxx.map
```

where xxxxx is the name of the font map. This updates your local (personal) font mappings. If you need to do this on a shared machine (eg in a lab or library), you need the Admin or root password, and use the command

```
updatemap-sys --force --enable Map=xxxxx.map
```

B.4.2 Updating font maps in MiKTeX

This procedure comes from the \TeX Users Group page at www.tug.org/fonts/fontinstall.html.

1. Edit the map configuration file `updatemap.cfg` in a DOS/Command Prompt window; type the command:

```
initexmf --edit-config-file updatemap
```

Edit this file in a text editor such as Notepad.

2. Add this one line to `updatemap.cfg` and save it:

```
Map newfont.map
```

3. Back at the DOS/Command prompt, type:

```
initexmf --mkmaps
```

Ignore any error messages.
Commands, errors, and viewing

Most people these days do their \LaTeX{}ing in a graphical windowing editor with menus, running in a modern operating system that uses windows, icons, fonts, and a mouse which moves a pointer. This probably works fine 95% of the time, when you’re dealing with one or two documents at a time, and everything you want to do is accessible through the menus, and you explicitly don’t want to see \LaTeX{} spilling its guts all over the place every time it reformats the document. Click here, move to there, cut, move somewhere else, paste, edit the text, write some more, click Typeset and you’re done.

However, life isn’t always that easy. Sometimes things go wrong, and you need to open up the lid and find out what it was. This appendix is a short description of how to run \LaTeX{} manually, via the command-line, instead of through your editor, and it also covers error messages, and a few internal details about viewing and printing.

The editor wasn’t always the primary interface to \TeX{}, except for actually writing and editing the document. Before editors with built-in \TeX{} controls became available, you had to leave your editor — or at least go to another window — and type a command to process your document, then another to view it or print it. For a small but significant number of people, running \LaTeX{} this way is still the order of the day.
APPENDIX C. COMMANDS, ERRORS, AND VIEWING

- Maybe they’re working on a remote mainframe or supercomputer console with no graphics, just a 3270 or VT-100 terminal like those in Figure C.1 on page 245;
- They might be using a smartphone where the editing facilities are limited and the scope for full menus entirely absent;
- Perhaps they are simply uninterested in all the bells and whistles of the modern interface, with too many menus doing things they can actually do faster typing instructions by hand;
- Possibly they’re using automation facilities that most \TeX{} editors don’t have, like the ability to apply the same edit to thousands of documents while you go and have a coffee or get on with something else;
- Or perhaps they are writing a system where \TeX{} is the embedded typesetter, so they’re actually working in a completely different scripting or programming language which does a lot of other things before calling on \TeX{} behind the scenes to do some typesetting.

Before I go any further I’m going to assume at this stage that you have typed a document (for example Figure 1.2 on page 7), and that you have saved it as a plaintext file with a filetype of `.tex` and a name of your own choosing, following the rules in the panel ‘Picking suitable filenames’ on p. 39.

C.1 Terminals and windows

Originally a terminal was a screen and a keyboard, looking very much like a standard desktop computer in the days before flat screens and windowing systems. There are still a surprising number of these around. The important point is that it was (is) a text-only interface to the computer. You got 25 lines of 80 fixed-width white-on-black or green-on-black characters, no fonts, no colours, and no mouse; maybe reverse-video as a form of highlighting (see Figure C.1 on page 245).

Nowadays the word usually means a ‘virtual terminal’: a window that behaves like a terminal — 25 lines of 80 fixed-width characters
in monochrome (see Figure C.2 on page 246). It’s a window into the heart of your computer. Even though you still have all your other windows visible, it knows nothing about them and can’t interact with them (except for copy and paste). But instead of being a padded cell, most terminals can do things many other windows can’t, like handling files in bulk, or to a schedule, or unattended, even forcing things to happen even when the graphical world outside has got itself jammed solid.

C.1.1 So where is the terminal window?

**Apple Macintosh OS X**: Click on [Finder > Applications > Utilities > Terminal];

**Microsoft Windows**: Click on the Windows or Start button, [All Programs > Accessories > Terminal] (in older versions it’s called [Command Prompt]);

**UNIX and GNU/Linux**: In most graphical interfaces, click on the menu [Applications > Accessories > Terminal] (in some systems it's called [Console]).

Figure C.1: Text-only display terminals

Images courtesy of Wikipedia. *Left*: IBM 3279 display by Retro-Computing Society of Rhode Island (CC BY-SA 3.0); *Right*: DEC VT100 terminal by Jason Scott (Flickr IMG_9976, CC BY 2.0), at the Living Computer Museum (apparently connected to the museum’s DEC PDP-11/70).
When you have finished using the terminal, it’s good practice to type \texttt{\$exit} (and press \textless{}\textgreater{}).

### C.1.2 Using the terminal window

On a physical terminal you usually have to log in first (very much like today: username and password). In a terminal window this isn’t usually necessary (see Figure C.2 on page 246).

**Figure C.2:** Virtual terminal in a window

In this example I’m logged into a computer called \texttt{nimrod} with my username \texttt{peter}. The system prompt is the directory name plus a dollar sign (the tilde indicates that I’m in my home directory system). For visibility, I underlined in red here the commands I typed, one to change to my \texttt{Documents} folder, and one to run \texttt{XeLaTeX} on the \texttt{quickstart.tex} document.

The first thing you see is the *prompt* (usually a dollar sign or percent sign, or maybe a greater-than pointer \texttt{C:\>} like MS-DOS used to use). When the prompt appears, you can type an instruction (command) and press the \textless{}\textgreater{} key at the end of the line to send it off to the computer for processing. Until you press \textless{}\textgreater{}, the computer has no idea you’ve finished typing: you MUST press \textless{}\textgreater{} at the end of each line of command for it to take effect.
C.1. TERMINALS AND WINDOWS

The results, if any, are displayed on the screen, and the prompt is displayed again ready for your next command. Some commands don’t have any output: if you change directory or delete a file, for example, you just get another prompt. There’s no message confirming the action, and no check to see if you really meant it. You said to do it, and it’s done.

C.1.3 How do I know what commands to type?

You have to learn or be told, or you have to look up what you want to do in a manual (or on the web). Terminals don’t have menus (there are simply far too many commands to fit in menus), and while UNIX and GNU/Linux systems (including Apple Macintosh OS X) have an online manual, it’s not searchable in the normal meaning of the word: you have to know in advance the name of the program command you want to look up.

Having said that, unless you plan on using the terminal for a lot of other work, there are really only half a dozen commands you need to be familiar with. The most important of these are the obvious ones: \$latex, \$pdflatex, \$xelatex, or \$latexmk; plus \$makeindex and \$biber or \$bibtex depending on whether or not you have an index, and how you want your bibliography formatted. Each command is the single-word short name of a program, and is followed by the name of the \LaTeX\ file you want to typeset.

A few other commands are pretty essential: they have nothing to do with \LaTeX\ — these are housekeeping or file-management commands and are built into every computer. There are a lot more of them but these are the ones you will likely need.

\$\texttt{cd \ Change Directory}, followed by the name of a folder or directory.

This makes sure you are in the right directory (folder) for what you want to do. Terminals always start up in your Home or Login directory, so you will almost always want to type this as your first command, eg

\begin{quote}
\$\texttt{cd Documents/reports/quarterly}
\end{quote}

Note that you have to know which directory you need to change into: the computer cannot know or guess. In Windows systems
you have to use the backslash instead of the normal slash to separate folders and filenames.

$ more This command shows you the contents of a file, screen by screen. It’s not an editor, so you can’t change the contents, but it’s a fast way to have a quick look at a small file without editing it. You press the spacebar to go to the next page, or the $ key to move line by line. Press $q$ to stop the program and get back to the command prompt.

```
$ more mythesis.aux
```

$ del or $ rm: DELete or ReMove, followed by the name of the file or folder you want to get rid of — $del is the Windows command and $rm is for all other desktop systems.

```
C:\Documents\reports\quarterly> del mythesis.aux
```

There is no check or safety-catch, and no Trash or Wastebasket to retrieve accidentally-deleted files from. When you delete a file, it’s really gone, immediately and forever.

$ ren or $ mv: REName or MoVe, followed firstly by the name of the file or folder you want to rename, and secondly by the name you want it renamed to — $ren is the Windows command and $mv is for all other desktop systems.

```
C:\Documents\thesis> ren thesis.tex book.tex
```

The $mv command renames a file by moving it to the new name; in fact a side-effect of its real purpose, which is to move files from one place to another, so both arguments can be whole file-paths:

```
```

$ exit Finishes your terminal session and closes the window.
C.2 Typesetting

Which \LaTeX command you type depends on what output you want and how you want it to be created — see the list on page xvii. Whichever way you run \LaTeX, it will process your file and display a log or record of what it’s doing (see Figure C.2 on page 246: it looks much the same no matter what system you use).

To typeset your document:

1. Make sure you are in the right directory (folder) in your terminal, then type the command ($\texttt{latex}$, $\texttt{pdflatex}$, or $\texttt{xelatex}$, for example; or $\texttt{latexmk}$ or one of the other workflow-management commands);

2. If you are using citation and reference commands for a bibliography, you will then need to run $\texttt{biber}$ or $\texttt{bibtex}$ (followed by a space and the name of your document), whichever you have chosen to use (see section 5.3.2.1 on page 112);

3. Run \LaTeX again as in item 1 on page 249 so that the citations are picked up;

4. If you are creating an index, you will then need to run $\texttt{makeindex}$ (followed by a space and the name of your document);

5. Run \LaTeX again as in item 1 on page 249 so that the citations and index references are resolved.

\LaTeX and all the ancillary programs write a transcript of what goes, and this will be shown in the window as well as being written into a log file.

If \LaTeX reports any errors — easily identifiable as lines in the log beginning with an exclamation mark (!) — \textit{don’t panic!} Turn to section C.3 on page 249, identify what went wrong, and fix it in your input file. Then re-run \LaTeX.

C.3 Errors and warnings

\LaTeX describes what it’s typesetting while it does it, and if it encounters something it doesn’t understand or can’t do, it will display a
APPENDIX C. COMMANDS, ERRORS, AND VIEWING

Exercise 17. Running \LaTeX{} in a terminal or console window

- Open a command window;
- Type `$cd$` followed by the name of the folder where you saved your sample document;
- Type your \LaTeX{} command followed by the name of your \LaTeX{} document.

message saying what’s wrong. It may also display warnings for less serious conditions.

Don’t panic if you see error messages: it’s very common for beginners as well as seasoned users to mistype or mis-spell commands, forget curly braces, type a forward slash instead of a backslash, or use a special character by mistake. Errors are easily spotted and easily corrected in your editor, and you can then run \LaTeX{} again to check you have fixed everything. Some of the most common errors are described in section C.3.1 on page 250 with an explanation of how to fix them.

Some editors show hotlinks in the \LaTeX{} log window where you can click on an error message and the cursor will jump to the line in your document where the error was spotted.

There is an extensive guide to how to handle errors in \LaTeX{} in (Beeton, 2017) (her presentation from TUG 2017) which also has a lot of useful information about how to work with \LaTeX{} in general.

C.3.1 Error messages

The format of an error message is always the same. Error messages begin with an exclamation message mark at the start of the line, and give a description of the error, followed by another line starting with the number, which refers to the line-number in your document file which
\LaTeX{} was processing when the error was spotted. Here’s an example, showing that the user mistyped the `\tableofcontents` command:

```latex
! Undefined control sequence.
1.6 \tableofcontents
```

When \LaTeX{} finds an error like this, it displays the error message and pauses. You must type one of the following letters to continue:

<table>
<thead>
<tr>
<th>Key</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>Stop immediately and exit the program.</td>
</tr>
<tr>
<td>q</td>
<td>Carry on quietly as best you can and don’t bother me with any more error messages.</td>
</tr>
<tr>
<td>e</td>
<td>Stop the program but re-position the text in my editor at the point where you found the error (this only works if you’re using an editor which \LaTeX{} can communicate with).</td>
</tr>
<tr>
<td>h</td>
<td>Try to give me more help.</td>
</tr>
<tr>
<td>i</td>
<td>(followed by a correction) means input the correction in place of the error and carry on (this is only a temporary fix to get the file processed. You still have to make that correction in the editor).</td>
</tr>
</tbody>
</table>

Some systems (\textit{Emacs} is one example) run \LaTeX{} with a ‘non-stop’ switch turned on, so it will always process through to the end of the file, regardless of errors, or until a limit is reached.

### C.3.2 Warnings

Warnings don’t begin with an exclamation mark: they are just comments by \LaTeX{} about things you might want to look into, such as overlong or underrun lines (often caused by unusual hyphenations, for example), pages running short or long, and other typographical niceties (most of which you can ignore until later).

Unlike other systems, which try to hide unevennesses in the text — usually unsuccessfully — by interfering with the letter-spacing, \LaTeX{} takes the view that the author or editor should be able to contribute. While it is certainly possible to set \LaTeX{}’s parameters so that the spacing is sufficiently sloppy that you will almost never get a warning about badly-fitting lines or pages, you will almost certainly just be delaying matters until you start to get complaints from your readers or publishers.
C.3.3 Examples

Only a few common error messages are given here: those most likely to be encountered by beginners. If you find another error message not shown here, and it’s not clear what you should do, ask for help.

Most error messages are self-explanatory, but be aware that the place where \TeX spots and reports an error may be later in the file than the place where it actually occurred. For example if you forget to close a curly brace which encloses, say, italics, \TeX won’t report this until something else occurs which can’t happen until the curly brace is encountered (eg the end of the document!) Some errors can only be righted by humans who can read and understand what the document is supposed to mean or look like.

Newcomers — remember to check the list of special characters: many errors when you are learning \TeX are due to accidentally typing a special character when you didn’t mean to. This disappears after a few hours as you get used to them.

C.3.3.1 Too many \'}s

\begin{verbatim}
! Too many \'}s.
1.6 \date{December 2004}
\end{verbatim}

The reason \TeX thinks there are too many \'}s here is that the opening curly brace is missing after the \texttt{\date} control sequence and before the word \texttt{December}, so the closing curly brace is seen as one too many (which it is!).

In fact, there are other things which can follow the \texttt{\date} command apart from a date in curly braces, so \TeX cannot possibly guess that you’ve missed out the opening curly brace — until it finds a closing one!

C.3.3.2 Undefined control sequence

\begin{verbatim}
! Undefined control sequence.
1.6 \dtae
{December 2004}
\end{verbatim}

In this example, \TeX is complaining that it has no such command (‘control sequence’) as \texttt{\dtae}. Obviously it’s been mistyped, but only
a human can detect that fact: all \LaTeX{} knows is that \texttt{dtae} is not a command it knows about — it’s undefined.

Mistypings are the commonest source of error. If your editor has drop-down menus to insert common commands and environments, use them!

### C.3.3.3 Runaway argument

```
Runaway argument?
{December 2004 \texttt{\texttt{maketitle}}
! Paragraph ended before \texttt{\texttt{date}} was complete.
<to be read again>
\par
1.8
```

In this error, the closing curly brace has been omitted from the date. It’s the opposite of the error in section C.3.3.1 on page 252, and it results in \texttt{\texttt{maketitle}} trying to format the title page while \LaTeX{} is still expecting more text for the date! As \texttt{\texttt{maketitle}} creates new paragraphs on the title page, this is detected and \LaTeX{} complains that the previous paragraph has ended but \texttt{\texttt{date}} is not yet finished.

### C.3.3.4 Capacity exceeded

```
! \TeX{} capacity exceeded, sorry [parameter stack size=5000].
```

This is rather more serious: it means \TeX{} has completely run out of memory. This will happen if you try to push the system too far, like getting it to read lines which are quite excessively long, or macros which are too complex to fit in memory (or possibly just badly-written). I had it happen once with an author who had written a single paragraph over 37 pages long. I suggested this was perhaps a style that was unfair on his readers...

### C.3.3.5 Underfull hbox

```
Underfull \texttt{hbox} (badness 1394) in \texttt{paragraph}
at lines 28--30
[]][LY1/brm/b/n/10 Bull, RJ: \texttt{LY1/brm/m/n/10}
Ac-count-ing in Busi-
[94]
```

Formatting Information
This is a warning that \LaTeX cannot stretch the line wide enough to fit, without making the spacing bigger than its currently permitted maximum. The \textit{badness} (0--10,000) indicates how severe this is (here you can probably ignore a badness of 1394). It says what lines of your file it was typesetting when it found this, and the number in square brackets is the number of the page onto which the offending line was printed.

The codes separated by slashes are the typeface and font style and size used in the line. Ignore them for the moment: details are in step 8 on page 238 if you're curious.

\subsection*{C.3.3.6 Overfull hbox}

\begin{quote}
[101]
Overfull \texttt{ hbox} (9.11617 pt too wide) in \texttt{paragraph}
\end{quote}

\begin{quote}
 at lines 860--861
\end{quote}

\begin{quote}
\texttt{[]/LY1/brm/m/n/10 \texttt{Windows, \texttt{LY1/brm/m/it/10 see}}}
\end{quote}

\begin{quote}
\texttt{\texttt{LY1/brm/m/n/10 \texttt{X Win-}}}
\end{quote}

And the opposite warning: this line is too long by a shade over 9pt. The chosen hyphenation point which minimises the error is shown at the end of the line (Win-). Line numbers and page numbers are given as before. In this case, 9pt is too much to ignore (over 3mm or more than $\frac{1}{8}''$), and a manual correction needs making (such as a change to the hyphenation), or the flexibility settings need changing (outside the scope of this book).

\subsection*{C.3.3.7 Missing package}

\begin{quote}
! \LaTeX Error: File \texttt{`paralisy.sty'} not found.
\end{quote}

\begin{quote}
Type \texttt{X} to quit or \texttt{<RETURN>} to proceed, 
\end{quote}

\begin{quote}
or enter new name. (Default extension: sty)
\end{quote}

Enter file name:

When you use the \texttt{\usepackage} command to request \LaTeX to use a certain package, it will look for a file with the specified name and the filetype \texttt{.sty}. In this case the user has mistyped the name of the \texttt{paralist} package, so it's easy to fix. However, if you get the name
right, but the package is not installed on your machine, you will need to download and install it before continuing (see Chapter 3 starting on page 55).

## C.4 Screen preview

Once you have typeset your document without errors (or even if there are still errors, but you want to see what it’s doing with them), most processors (\TeX, PDF\TeX{}, etc) will have created a PDF file and most editors will pop up a PDF viewer so you can see what the results are. In this case you can skip to section C.4.3 on page 257.

However, standard (‘old’) \TeX{} will have created a DVI (DeVeice-Independent) file. This was \TeX{}’s original format from the days before PDF and Postscript, but it is still popular with some users. Again, most editors will pop up a DVI viewer so you can see what the results are, but if you are working from the command-line, you need to run it yourself.

### C.4.1 Previewing DVI output

To see the typeset output from the command-line, type the name of a suitable DVI viewer followed by the name of your document (without the filetype extension). Common viewers are yap (Windows) and xdvii (UNIX and GNU/Linux systems, including Apple Macintosh OS X; on a Mac you need to install \textit{XQuartz} to run xdvii). A WYSIWYG preview window will appear with your typeset display (see Figure C.3 on page 256).

Most previewers have a wide range of scaling, zooming, and measuring functions, but as this is a picture of your output, you cannot edit the image. To change it, you edit your source document and reprocess it.

With xdvii and its derivatives like dviview, you can leave the display window open, and after you’ve reprocessed your document through \TeX{}, moving your mouse back into the window will make the display update automatically (click your mouse if your windowing system needs a click to focus).

Figure C.3 on page 256 shows xdvii displaying a page. With a standard three-button mouse you get three levels of micro-zoom to let you inspect fine details.
APPENDIX C. COMMANDS, ERRORS, AND VIEWING

Figure C.3: DVI preview

C.4.2 Previewing with PostScript

PostScript is a page description programming language invented by Adobe and used in laser printers and high-end typesetters. It’s been the universal standard for electronically-formatted print files for nearly two decades, and all printers and publishers have been accustomed to using it. PDF is a descendant of PostScript, and as largely taken over, but PostScript itself is still quite common, partly because it is very robust, and is usually an ASCII file, which makes it very portable and easy to generate. The drawback is the large size of PostScript files, especially if they contain bitmapped graphics.

The dvips program which comes with all TeX distributions is used to generate PostScript files directly from your DVI output. These .ps files can be viewed, printed, sent to a platemaker or filmsetter, put online for downloading, or converted to PDF.
DVI viewers cannot render some PostScript graphical manipulations like rotating and deforming, so an alternative to viewing the DVI file direct is to generate a PostScript file and use a PostScript viewer. You may have to do this for your publisher anyway, and many editors can be configured to do this by default. Look for a dvips toolbar icon or menu entry and click on it.

It’s also very simple to do from the command-line: let’s assume your \LaTeX file was called quickstart.tex, so processing it has created quickstart.dvi. Just type

```
dvips -o quickstart.ps quickstart
```

in a command window, and dvips will create quickstart.ps which can be used both for previewing and printing, or converted into or out of PDF with the ps2pdf and pdf2ps utility programs which should have been installed automatically along with your \TeX system.

To view a PostScript file, you need a PostScript previewer like GSview, which works with the PostScript interpreter Ghostscript, which should also have been installed automatically along with your \TeX system (if not, install both now: GSview is separately licensed and cannot legally be included in some older \TeX distributions, so you may have to download it yourself).

Like xdvi, GSview can be set to watch the PostScript file and automatically update the display any time the file is changed, without you having to click on the window.

### C.4.3 Previewing with PDF

The PDF is a derivative of PostScript. Whereas PostScript is a real programming language in itself, PDF is in effect the result of processing a document through PostScript: it’s a binary file format, extremely compact, and well-supported on all platforms.

If your system is configured to generate PDF files direct instead of DVI files, just open the .pdf file using any PDF viewer by typing the name of the viewer followed by the name of your document (without the filetype extension).

Adobe’s Acrobat Reader cannot automatically update its display in the way that xdvi and GSview can, when you reprocess your document. You have to close the display with Ctrl+W and reload the file with Alt...
If you are using it direct from within your editor instead, this closing and opening should be done for you.

**Bitmap preview fonts in Acrobat Reader**

Acrobat Reader is poor at rendering Type 3 bitmap fonts. If you are using these you will see a very fuzzy display at low magnifications. It will print perfectly, but Acrobat Reader’s display is disappointing. The solution is to use a better previewer or to upgrade to the Type 1 versions of the fonts if possible, or both. If you need to use Type 3 fonts in PDFs, you probably need to warn your readers to expect a fuzzy display from Acrobat Reader (but good printout), and to change to a better reader if they can.

### C.5 Printer output

\TeX{} systems print on almost anything from portable and domestic dot-matrix printers through normal office ink-jet and laser printers up to the biggest commercial phototypesetters, including a host of other devices in between (numerically-controlled stencil-cutters, knitting machines, and ink-jet cake-decorators, to name but a few).

When using a previewer, printing works in the normal manner through your system’s printer drivers: you just click on the Print icon in your preview. However, \TeX{}’s ancillary programs are capable of creating highly-optimised printfiles for almost any printer or typesetter, allowing you to send printout to printers that are not connected to your computer.

The rest of this section deals with how to print on older systems without a print management converter, where the procedure may vary slightly according to how you do your typesetting and previewing.

**If you are using DVI**: and you have a previewer which has a Print function configured for your printer, you can use that. If not, create a PostScript file and use GSview instead.

**If you are using PDF**: you can print directly from your PDF viewer. Be careful about using Adobe Acrobat Reader’s ‘Shrink to fit’
C.5. PRINTER OUTPUT

option, as it will change the size of your document so all your measurements will be different. Turn it off.

**Non-PostScript printers**: You can create a *PostScript* file with *dvips* (see section C.4.2 on page 256) and use *GSview* to print it (*GSview* can print *PostScript* files to almost any make or model of non-*PostScript* printer).

**If you have a real *PostScript* printer**: or you are using a system with built-in *PostScript* printing support (such as Linux or Mac), you can create and send *PostScript* output directly from your editor to the printer without the need to open it in a previewer first. In *Emacs*, for example, this is what happens when you use the menu item.

Both the *dvips* program and all the previewers that print tend to have facilities for printing selected pages, printing in reverse, scaling the page size, and printing only odd or even pages for two-sided work. If you are using *PostScript* there are programs for manipulating the output (*pstop*), for example to perform page imposition to get 4, 8, or 16 pages to a sheet for making booklets (*psnup*).

**Exercise 18. Print it!**

Show that you have understood the process of typesetting, previewing, and printing, by displaying your document and printing it.

If you need a non-*PostScript/Ghostscript* solution, install a separate *TeX* print driver for your printer. Some may be supplied with your *TeX* installation, and there are dozens more on CTAN. Their names all start with *dv* and are followed by an abbreviation for the printer make or model like *dvieps* for Epson, *dvihp* for Hewlett-Packard, *dvialw* for Apple LaserWriters, etc. Configure the driver to print directly to the print queue, or pipe it to the print queue manually. On Linux with an HP printer, for example, this would be

```
dvihp quickstart | lpr
```
Microsoft Windows has no easy way to bypass the print spool, but you can do it from an MS-DOS command window with (using a HP printer as an example):

```
dvihp quickstart -o quickstart.hp
copy /b quickstart.hp LPT1:
```

Read the documentation for the driver, as the options and defaults vary.
User Groups

The \TeX{} Users Group (TUG) was founded in 1980 for educational and scientific purposes: to provide an organisation for those who have an interest in typography and font design, and are users of the \TeX{} typesetting system invented by Donald Knuth. TUG is run by and for its members and represents the interests of \TeX{} users worldwide. There are many regional and sectoral user groups organised on a geographic, language, or topical basis: see the list at \url{www.tug.org/usergroups.html} for details.

D.1 Conferences and meetings

TUG and many other user groups hold annual meetings where members, newcomers, and visitors can hear about new features, discuss developments, and learn about \TeX{}, typography, and related topics. Full details of TUG-sponsored meetings are on the TUG web site.

This list is maintained by the author, based on public announcements in mailing lists, web forums, and newsgroups. If you have details of a meeting which is not shown here, please contact me.

**GuIT Annual Conference** : The XIII Italian \TeX{} User Group Conference was held at the Catholic University of Brescia;

**MarkupUK 2019** : MarkupUK, which was new in 2018, replaces the event formerly known as XML London. Exact dates and location...
APPENDIX D. USER GROUPS

are yet to be announced. Although primarily related to XML, many aspects of markup are closely related to the use of \LaTeX and the handling of structured documents. Abstract submissions can be sent to submit@markupuk.org.

**Markup UK 2018**: A new event in 2018, Markup UK took place in London on June 9–10, 2018, in the Skempton Building of Imperial College’s South Kensington campus.

**TEI Conference 2018**: The 2018 TEI Consortium Meeting ‘TEI as a Global Language’ will be held in Tokyo on September 9–13 in conjunction with the annual conference of the Japanese Society for Digital Humanities.

**TEI Conference**: The 2016 TEI Conference took place at the Austrian Academy of Sciences in Vienna, hosted by the Austrian Centre of Digital Humanities from 26–30 September 2016.

**TEI Conference**: The 2017 TEI Consortium Meeting was held in Victoria, BC on November 11–15.

**Digital Humanities**: The 2017 Digital Humanities Conference took place August 8–11 in Montréal, co-hosted by McGill University and the Université de Montréal. This was the week after Balisage.

**Digital Humanities 2018**: The Digital Humanities Conference 2018 will be held in Mexico City, hosted by El Colegio de México and the Universidad Nacional Autónoma de México (UNAM) on June 26–28, in alliance with the Red de Humanidades Digitales (date to be announced).

**European Association for Digital Humanities 2018**: The European Association for Digital Humanities will hold its inaugural annual conference, on the theme *Data in Digital Humanities* at the National University of Ireland, Galway from 7–9 December 2018.

**BachoTEX 2018**: The 26th annual Bacho\TeX 2018 conference of GUST will be held April 28th–May 2nd, at the usual location of Bachotek near Brodnica, in the north-east of Poland.
D.1. CONFERENCES AND MEETINGS

Bacho\TeX\ 2016: Convergence: \TeX, get out of the closet!: The 24th yearly GUST Bacho\TeX\ conference in 2016 was held from April 29th to May 3rd, at the usual place of Bachotek near Brodnica, in the north-east of Poland.

TUG 2019: The \TeX\ Users Group 2019 conference will take place August 9–11 (Fri-Sun), 2019, at the Sheraton Hotel in Palo Alto, California (San Francisco Bay Area). Web registration will be available later.

TUG 2018: The \TeX\ Users Group 2018 conference took place July 20–22 as an official satellite conference of the International Congress of Mathematicians in Rio de Janeiro, Brazil.

TUG 2017: The \TeX\ Users Group 2017 conference was a joint meeting with GUST to celebrate the 25th anniversary of Bacho\TeX, and took place in Bachotek, Poland, on April 29–May 3, 2017.

TUG 2016: covering the \TeX\ world: The main annual conference was in Toronto, Canada, from 25–27 July, co-sponsored by DANTE e.V., the German-speaking \TeX\ users group.

JabCon 2016: for JabRef users: The JabRef conference for users and developers took place in Vienna, Austria from 25-26 January.

Con\TeXxt\ 2017: Con\TeXxt\ Gardening: The 11th annual Con\TeXxt\ meeting will be held in Butzbach-Maibach, Germany, from 11–17 September 2017. This meeting will give Con\TeXxt\ and Lua\TeX\ developers as well as users the chance to present results, experiences, and ideas on future development. The talks will be followed by tutorials on Con\TeXxt\ and Lua\TeX\ techniques.

Balisage: The Balisage markup conference is the principal technical meeting specifically about markup, including both XML and \TeX. The 2019 meeting will be held in Rockville MD July 30th to August 2nd preceded by a one-day Symposium (topic to be announced) on July 29th.

Balisage 2018: The Balisage markup conference 2018 was held in Rockville, MD, on July 30–August 3. Proceedings are now online at www.balisage.net/Proceedings/index.html.
APPENDIX D. USER GROUPS

Balisage: The Balisage markup conference 2017 was held in Rockville, MD, on August 1–4. Proceedings are now online at www.balisage.net/Proceedings/index.html. This was the week before Digital Humanities in Montréal.

XML Summerschool: Although this isn’t a \TeX event as such, many \LaTeX users also use XML, and many XML users also use \LaTeX, so the annual XML Summer School may be of interest. This will be held in St Edmund Hall, Oxford in September 2018. It’s week-long event covers everything from an introduction for the beginner up to XML in publishing, transformation with XSLT2 and XQuery, and the use of Linked Data.

D.2 TUG membership benefits

Members of TUG help to support and promote the use of \TeX, META-\FONT, and related systems worldwide. All members receive ‘TUGboat’, the journal of the \TeX Users Group, the \TeX Live software distribution (on DVD), and the CTAN software distribution (containing most of the CTAN archive).

In addition, TUG members can vote in TUG elections, and receive discounts on annual meeting fees, store purchases, and TUG-sponsored courses. TUG membership (less benefits) is tax-deductible, at least in the USA. See the TUG Web site for details.

D.3 Becoming a TUG member

Please see the forms and information at www.tug.org/join.html. You can join online, or by filling out a paper form. The NTG (Dutch) and UKTUG (United Kingdom) \TeX user groups have joint membership agreements with TUG whereby you can receive a discount for joining both TUG and the local user group. To do this, please join via www.ntg.nl/newmember.html (the NTG membership page) or uk.tug.org/Membership/ (the UKTUG page), respectively, and select the option for joint membership.

Each year’s membership entitles you to the software and TUGboat produced for that year (even if it is produced in a subsequent calendar year, as is occasionally the case). You can order older issues of
TUGboat and \TeX\ memorabilia through the TUG store (www.tug.org/store).

The current (2015) TUG membership fee before March 31st is $85 (US) per year for individuals and $55 for students, new graduates, seniors, and citizens of countries with modest economies. Add $20 to the membership fee after March 31 to cover additional shipping and processing costs. The current rate for non-voting subscription memberships (for libraries, for example) is $105. The current institutional rate is $500, which includes up to seven individual memberships.

D.4 Privacy

TUG uses your personal information only to mail you products, publications, notices, and (for voting members) official ballots. Also, if you give explicit agreement, we may incorporate it into a membership directory which will be made available only to TUG members.

TUG neither sells its membership list nor provides it to anyone outside of its own membership.
The ASCII character set

The American Standard Code for Information Interchange (ASCII) was invented in 1963, and after some development settled down in 1984 as standard X3.4 of American National Standards Institute (ANSI). It represents the 95 codes for the printable characters (A-Z, a-z, 0-9, and punctuation) of the unaccented Latin alphabet, plus 33 internal ‘control characters’ originally intended for the control of computers, programs, and external devices like printers, screens, disks, modems, etc.

Many other character sets (strictly speaking, ‘character repertoires’) have been used for accented Latin characters and for other (non-Latin) writing systems, for representing the symbols people use when writing text on computers, but the current standard is ISO 10646 (Unicode), which covers pretty much all the marks the human race makes when communication, and I strongly recommend you use only Unicode UTF-8 when writing for \LaTeX{}.

However, most programs and computers use ASCII internally for all their coding, the exceptions being XML-based languages like XSLT, which default to UTF-8, but are inherently designed to be usable with any writing system; and a few specialist systems like APL.

Although the \TeX{} and \LaTeX{} file formats can easily be used with many other encoding systems (see the discussion of the \texttt{inputenc} package in section 1.8 on page 19), their markup is based on ASCII. It is therefore important that you know where to find all 95 of the printable characters, as some of them are not often used in other text-formatting systems.
APPENDIX E. THE ASCII CHARACTER SET

The following table shows all 128 characters, with their decimal, octal (base-8), and hexadecimal (base-16) code numbers.

Decimal number values are under or beside each character. The index numbers in the first and last columns are for finding the octal (base-8) and hexadecimal (base-16) values respectively. Replace the arrow with the number or letter from the top row label of the column (if the arrow points up) or from the bottom row label of the column (if the arrow points down).

Example: The Escape character (ESC) is decimal 27; octal '033 (03 for the row, 3 for the number at the top of the column because the arrow points up), and hexadecimal ”1B (1 for the row, B for the letter at the bottom of the column because the arrow points down).
Table E.1: The ASCII characters

<table>
<thead>
<tr>
<th>Oct</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>'00'</td>
<td>NUL</td>
<td>SOH</td>
<td>STX</td>
<td>ETX</td>
<td>EOT</td>
<td>ENQ</td>
<td>ACK</td>
<td>BEL</td>
<td>&quot;0↑&quot;</td>
</tr>
<tr>
<td>'01'</td>
<td>BS</td>
<td>HT</td>
<td>LF</td>
<td>VT</td>
<td>FF</td>
<td>CR</td>
<td>SO</td>
<td>SI</td>
<td>&quot;0↓&quot;</td>
</tr>
<tr>
<td>'02'</td>
<td>DLE</td>
<td>DC1</td>
<td>DC2</td>
<td>DC3</td>
<td>DC4</td>
<td>NAK</td>
<td>SYN</td>
<td>ETB</td>
<td>&quot;1↑&quot;</td>
</tr>
<tr>
<td>'03'</td>
<td>CAN</td>
<td>EM</td>
<td>SUB</td>
<td>ESC</td>
<td>FS</td>
<td>GS</td>
<td>RS</td>
<td>US</td>
<td>&quot;1↓&quot;</td>
</tr>
<tr>
<td>'04'</td>
<td>&quot; &quot;</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
<td>'</td>
<td>(</td>
<td>)</td>
<td>&quot;2↑&quot;</td>
</tr>
<tr>
<td>'05'</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>.</td>
<td>/</td>
<td></td>
<td></td>
<td></td>
<td>&quot;2↓&quot;</td>
</tr>
<tr>
<td>'06'</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>&quot;3↑&quot;</td>
</tr>
<tr>
<td>'07'</td>
<td>8</td>
<td>9</td>
<td>:</td>
<td>;</td>
<td>&lt;</td>
<td>=</td>
<td>&gt;</td>
<td>?</td>
<td>&quot;3↓&quot;</td>
</tr>
<tr>
<td>'10'</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>&quot;4↑&quot;</td>
</tr>
<tr>
<td>'11'</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
<td>O</td>
<td>&quot;4↓&quot;</td>
</tr>
<tr>
<td>'12'</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
<td>W</td>
<td>&quot;5↑&quot;</td>
</tr>
<tr>
<td>'13'</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>[</td>
<td>]</td>
<td>^</td>
<td>_</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'14'</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td></td>
<td>&quot;6↑&quot;</td>
</tr>
<tr>
<td>'15'</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
<td>n</td>
<td>o</td>
<td>&quot;6↓&quot;</td>
</tr>
<tr>
<td>'16'</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
<td>&quot;7↑&quot;</td>
</tr>
<tr>
<td>'17'</td>
<td>x</td>
<td>y</td>
<td>z</td>
<td>{</td>
<td>}</td>
<td>~</td>
<td>DEL</td>
<td></td>
<td>&quot;7↓&quot;</td>
</tr>
</tbody>
</table>

| 8  | 9  | A  | B  | C  | D  | E  | F  | 271 |
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

F.1 PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document ‘free’ in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of ‘copyleft’, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals;
it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

F.2 APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The ‘Document’, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as ‘you’. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A ‘Modified Version’ of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A ‘Secondary Section’ is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The ‘Invariant Sections’ are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The ‘Cover Texts’ are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.
A ‘Transparent’ copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not ‘Transparent’ is called ‘Opaque’.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, \LaTeX{} input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The ‘Title Page’ means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, ‘Title Page’ means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The ‘publisher’ means any person or entity that distributes copies of the Document to the public.

A section ‘Entitled XYZ’ means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as ‘Acknowledgements’, ‘Dedications’, ‘Endorsements’, or ‘History’.) To ‘Preserve the Title’ of such a section when you modify the Document means that it remains a section ‘Entitled XYZ’ according to this definition.
The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

F.3 VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

F.4 COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.
F.5. MODIFICATIONS

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

F.5 MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the
APPENDIX F. LICENSE

Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C State on the Title page the name of the publisher of the Modified Version, as the publisher.

D Preserve all the copyright notices of the Document.

E Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.

H Include an unaltered copy of this License.

I Preserve the section Entitled ‘History’, Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled ‘History’ in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the ‘History’ section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K For any section Entitled ‘Acknowledgements’ or ‘Dedications’, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M Delete any section Entitled ‘Endorsements’. Such a section may not be included in the Modified Version.

N Do not retitle any existing section to be Entitled ‘Endorsements’ or to conflict in title with any Invariant Section.

O Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled ‘Endorsements’, provided it contains nothing but endorsements of your Modified Version by various parties — for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.
F.6 COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled ‘History’ in the various original documents, forming one section Entitled ‘History’; likewise combine any sections Entitled ‘Acknowledgements’, and any sections Entitled ‘Dedications’. You must delete all sections Entitled ‘Endorsements’.

F.7 COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.
F.8 AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an ‘aggregate’ if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

F.9 TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled ‘Acknowledgements’, ‘Dedications’, or ‘History’, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.
APPENDIX F. LICENSE

F.10 TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

F.11 FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See Copyleft.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License ‘or any later version’ applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the
Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

F.12 RELICENSING

‘Massive Multiauthor Collaboration Site’ (or ‘MMC Site’) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A ‘Massive Multiauthor Collaboration’ (or ‘MMC’) contained in the site means any set of copyrightable works thus published on the MMC site.

‘CC-BY-SA’ means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

‘Incorporate’ means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is ‘eligible for relicensing’ if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

F.13 ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright © YEAR YOUR NAME
APPENDIX F. LICENSE

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled ‘GNU Free Documentation License’.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the ‘with…Texts.’ line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.
References

Providence, RI: AMS.

Anon. (1100). Táin bó Cúailnge. In *Leabhar na h-Uidhri: The Book of the
Dun Cow* (p. 55). Dublin: Royal Irish Academy [1870].

Beeton, B. (2017). Debugging \TeX files: Illegitimi non carborundum.

Identifier (URI): Generic Syntax*. Reston, VA: Internet Engineering
Task Force.

Berry, K. (2009). *Fontname: Filenames for \TeX fonts*. Portland, OR: \TeX
Users Group.

Bradner, S. (1997). *Key words for use in RFCs to Indicate Requirement

Alignment. In L. Burnard & M. Sperberg-McQueen (Eds.),
*Guidelines for the Text Encoding Initiative* (Chap. 17). Oxford: OUP.

Carnes, L. & Berry, K. (Eds.). (2004). The Prac\TeX Journal, Portland,
OR: \TeX Users Group.

the author.

Portland, OR: \TeX Users Group.

Portland, OR: \TeX Users Group.


Flynn, P. (2001). \TeX — a mass market product?: Or just an image in
need of a makeover? In *Tug 2001* (Vol. 22, 3). Portland, OR: \TeX
Users Group.

Flynn, P. (2002). Formatting Information. In *TUGboat* (Vol. 23, 2,
pp. 115–250).

References


comp.text.tex,
(MPG.18d82140d65ddc5898968c@news.earthlink.net), (all pages).


References

Oetiker, T. (2014). *A (Not So) Short Introduction to \TeX\ 2\epsilon*. Portland, OR: \TeX\ Users Group.
Reckdahl, K. (2006). *Using imported graphics in \TeX\ 2\epsilon*. \TeX\ Users Group.
Talbot, N. (2014). *Creating a \TeX\ Minimal Example*. Portland, OR: \TeX\ Users Group.
\TeX\ Users Group. (2003). *Getting Started with \TeX, \LaTeX, and friends*. Portland, OR: \TeX\ Users Group.
Index

See Table 2 on page xxx for the meanings of the typographic formatting used here and in the text.

\", 22
\', 22
\{, (31, 32
\}, 31, 32
\-, 26
\., 22
/\rho, 236
\=, 22
\&, 162, 164
\[, 32
\], 32
\~, 21, 22
\', 22
\', 21, 22
\, 22

\textbf, 17
\textbf, 254

Array

Accents

A4paper, 36, 38
\AA, 22
\aa, 22

AbiWord, 175, 177, 178
abstract, 44, 45
\abstractname, 45
abstracts, 44
accents, 19

Acrobat Reader, 185, 194, 222, 258
acronym, 126
\addbibresource, 120
\addcontentsline, 52
\addtocontents, 53
AE, 22

\ae, 22
\affiliation, 42
afm2tfm, 234
textrtf, 17
Apple Mac
editor, 189, 194
installation, 189
apt, 190
apt-get, xxvi
Aquamac, 222
\arabic, 170
\arctinfo, 96
textrtf, 13
array, 85
\arraybackslash, 85
\arraystretch, 87
\article, 11, 35
\author, xxx, 42–44, 108, 162
AutoCAD, 95
avant, 143
\b, 100
\b, 22
babel, 20, 50, 120
textrtf, 9
textrtf, 254
BaKoMa\ TeX, xxix
\baselinestretch, 133
bbding, 73, 171
beer, 123

\textbf, 13

American, 123
Rogue
\begin{latex}

\textbf{Chocolate Stout, 124}
\begin{itemize}
  \item \textit{begin}, 20, 40, 41, 45, 74, 82, 163, 180
  \item \textit{bfseries}, 12, 149
  \item \textit{biber}, ix, 5, 19, 62, 112–114, 120–122, 125
  \item \textit{biber}, 247, 249
  \item \textit{BIBINPUTS}, 116
  \item \textit{biblatex}, 5, 112, 113
  \item \textit{biblatex}, ix, 5, 19, 112–114, 116–121
  \item \textit{biblatex-apa}, 114
  \item \textit{biblatex-cheatsheet}, 114
  \item \textit{biblatex-chicago}, 114
  \item \textit{biblatex}, 5, 112, 113
  \item \textit{biblatex}, ix, 5, 19, 112–114, 116–121
  \item \textit{biblatex-apal}, 114
  \item \textit{biblatex-chestsheets}, 114
  \item \textit{biblatex-chicago}, 114
  \item \textit{biblatex}, 5, 112, 113
  \item \textit{biblatex}, ix, 5, 19, 112–114, 116–121
  \item \textit{biblatex}, 125
  \item \textit{biblatex}, 247, 249
  \item \textit{bigskip}, 132
  \item \textit{textbf}, 2, 229
  \item \textit{book}, 35
  \item \textit{book}, 47
  \item \textit{bookman}, 143
  \item \textit{boxes}, 99
  \item \textit{bp (big points)}, 25
  \item \textit{c}, 22
  \item \textit{c}, 22
  \item \textit{Calibre}, 2
  \item \textit{caption}, 81, 89, 111
  \item \textit{cat}, 237
  \item \textit{cc (Giceros)}, 25
  \item \textit{cd}, 66, 250
  \item \textit{center}, 11, 12, 29, 87, 88, 148
\end{itemize}
\end{latex}
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>\abstractname, 45</td>
<td></td>
</tr>
<tr>
<td>\addbibresource, 120</td>
<td></td>
</tr>
<tr>
<td>\addcontentsline, 52</td>
<td></td>
</tr>
<tr>
<td>\addtocontents, 53</td>
<td></td>
</tr>
<tr>
<td>\AE, 22</td>
<td></td>
</tr>
<tr>
<td>\ae, 22</td>
<td></td>
</tr>
<tr>
<td>\affiliation, 42</td>
<td></td>
</tr>
<tr>
<td>\arabic, 170</td>
<td></td>
</tr>
<tr>
<td>\arraybackslash, 85</td>
<td></td>
</tr>
<tr>
<td>\arraystretch, 87</td>
<td></td>
</tr>
<tr>
<td>\author, xxx</td>
<td></td>
</tr>
<tr>
<td>\bfseries, 12, 149</td>
<td></td>
</tr>
<tr>
<td>\biber, 247, 249</td>
<td></td>
</tr>
<tr>
<td>\bibliography, 121</td>
<td></td>
</tr>
<tr>
<td>\bibliographystyle, 117, 121</td>
<td></td>
</tr>
<tr>
<td>\bibtex, 247, 249</td>
<td></td>
</tr>
<tr>
<td>\bigskip, 132</td>
<td></td>
</tr>
<tr>
<td>\c, 22</td>
<td></td>
</tr>
<tr>
<td>\citation, 81, 89, 111</td>
<td></td>
</tr>
<tr>
<td>\cat, 237</td>
<td></td>
</tr>
<tr>
<td>\cd, 66, 250</td>
<td></td>
</tr>
<tr>
<td>\captioning, 29, 85, 87, 148</td>
<td></td>
</tr>
<tr>
<td>\chapter, 47, 48</td>
<td></td>
</tr>
<tr>
<td>\cite, 112, 117-119, 121, 122</td>
<td></td>
</tr>
<tr>
<td>\citep, 118, 121, 122</td>
<td></td>
</tr>
<tr>
<td>\citet, 118, 121</td>
<td></td>
</tr>
<tr>
<td>\clearpage, xxx</td>
<td></td>
</tr>
<tr>
<td>\cline, 84</td>
<td></td>
</tr>
<tr>
<td>\color, 58, 155</td>
<td></td>
</tr>
<tr>
<td>\colorbox, 102, 156</td>
<td></td>
</tr>
<tr>
<td>\d, 22</td>
<td></td>
</tr>
<tr>
<td>\date, 42, 44, 162, 252, 253</td>
<td></td>
</tr>
<tr>
<td>\datesubmitted, 42</td>
<td></td>
</tr>
<tr>
<td>\DeclareFontFamily, 239</td>
<td></td>
</tr>
<tr>
<td>\DeclareFontShape, 239, 240</td>
<td></td>
</tr>
<tr>
<td>\def, 167</td>
<td></td>
</tr>
<tr>
<td>\definecolor, 155, 156</td>
<td></td>
</tr>
<tr>
<td>\del, 248</td>
<td></td>
</tr>
<tr>
<td>\documentclass, 20, 34, 38, 45, 57, 152, 180</td>
<td></td>
</tr>
<tr>
<td>\doublespacing, 133</td>
<td></td>
</tr>
<tr>
<td>\dvips, 97, 193</td>
<td></td>
</tr>
<tr>
<td>\EF, 162</td>
<td></td>
</tr>
<tr>
<td>\emph, 154, 180</td>
<td></td>
</tr>
<tr>
<td>\end, 11, 39, 40, 45, 105, 180</td>
<td></td>
</tr>
<tr>
<td>\enspace, 134</td>
<td></td>
</tr>
<tr>
<td>\EUR, 17, 85</td>
<td></td>
</tr>
<tr>
<td>\exit, 139, 246</td>
<td></td>
</tr>
<tr>
<td>\fancyhead, 136</td>
<td></td>
</tr>
<tr>
<td>\fbox, 90, 101, 102, 156</td>
<td></td>
</tr>
</tbody>
</table>
\fc-cache, 139  
\fc-list, 230  
\fcolorbox, 102, 156  
\flushleft, 148  
\flushright, 148  
\fnsymbol, 108  
\fontencoding, 147  
\fontfamily, 143, 148  
\fontsize, 152  
\footcite, 118, 119  
\footnote, xxx, 103, 107, 108  
\footnotesize, 152  
\foreign, 154, 165  
\foreignlanguage, 30  
\gloss, 125  
\glossary, 125  
\gls, 125  
\graphicspath, 96, 97  
\grep, 230  
\H, 22  
@Heading, 3  
@h1, 3  
@h1, 3  
@l, 22  
@l, 22  
\label, 77, 81, 89, 110–112  
\labelitemi, 171  
\labelitemiv, 172  
\LARGE, 152  
\Large, 12, 152  
\large, 152  
\LaTeX, 179  
\latex, 8, 62, 193, 247, 249  
\latexmk, 247, 249  
\leftmark, 137  
\linebreak, 166  
\listoffigures, 52  
\listoftables, 52  
\ls, 227  
\lstinline, 104, 106  
\makeatletter, 163  
\makeatother, 163  
\makeglossary, 124  
\makeindex, 123  
\makeindex, 124, 247, 249  
\maketitle, 42–45, 52, 58, 147, 162, 163, 253  
\man, 124  
\marginal, 109  
\markboth, 135  
\markright, 135  
\mbox, 27, 166  
\medskip, 132  
\mkdir, 217, 218, 231, 236  
\multicolumn, 86, 88  
\mv, 248  
\newcommand, 162, 167  
\newcounter, 78  
\newfontface, 146  
\newfontfamily, 146  
\newgeometry, 132

Formatting Information
\textbf{INDEX}

\begin{itemize}
  \item \texttt{d}, 22
  \item \texttt{dash}, 28
    \begin{itemize}
      \item long, 28
      \item short, 28
    \end{itemize}
  \item \texttt{datatool}, 81, 86
  \item \texttt{date}, 42, 44, 162, 252, 253
  \item \texttt{datesubmitted}, 42
    \begin{itemize}
      \item \texttt{DCF}, 3
      \item \texttt{dcolumn}, 86
      \item \texttt{dd (Didot points)}, 25
    \end{itemize}
  \item \texttt{DeclareFontFamily}, 239
  \item \texttt{DeclareFontShape}, 239, 240
  \item \texttt{def}, 167
  \item \texttt{definecolor}, 155, 156
  \item \texttt{del}, 248
  \item \texttt{texbf}, 248
  \item \texttt{description}, 74, 75
  \item \texttt{description*}, 75
  \item \texttt{detex}, 186
  \item \texttt{texbf}, 49
  \item \texttt{dimensions}, 23
  \item \texttt{DOCTYPE}, 61
  \item \texttt{DocBook}, 177
  \item \texttt{DocBook 5}, xxxi
  \item \texttt{document}, xxiii, 11, 12, 20, 41, 42
  \item \texttt{document class}, 34
  \item \texttt{documentclass}, 20, 34, 38, 45, 57, 152, 180
  \item \texttt{Dolphin}, 62, 157, 217
  \item \texttt{double-spacing}, 133
  \item \texttt{doublespacing}, 133
  \item \texttt{draft}, 37
  \item \texttt{Draw}, 92, 96
  \item \texttt{duerer}, 142
  \item \texttt{dvialw}, 260
  \item \texttt{dvips}, 260
  \item \texttt{dvihp}, 260
  \item \texttt{dvips}, 93, 97, 257, 259
  \item \texttt{dvips}, 97, 193
  \item \texttt{dvipsnames}, 155
  \item \texttt{dviview}, 255
  \item \texttt{DynaTag}, 176
  \item \texttt{\texttt{EF}}, 162
  \item \texttt{eia d}, 142
  \item \texttt{El Capitan}, 194
  \item \texttt{em (relative measure)}, 25
  \item \texttt{Emacs}, xxvi, 10, 18, 86, 87, 222–224, 251, 259
  \item \texttt{emacs}, 138
  \item \texttt{\texttt{emph}}, 154, 180
  \item \texttt{empty}, 135
  \item \texttt{end}, 11, 39, 40, 45, 105, 180
  \item \texttt{endnote}, 108
  \item \texttt{enspace}, 134
  \item \texttt{enumerate}, 73
  \item \texttt{enumerate*}, 75, 76
  \item \texttt{enumi}, 78
  \item \texttt{enumii}, 78
  \item \texttt{enumiii}, 78
  \item \texttt{enumitem}, 65, 74, 75, 77, 132
  \item \texttt{enumiv}, 78
  \item \texttt{environment}, 72
  \item \texttt{environment}, xxx
  \item \texttt{texbf}, 11, 40
\end{itemize}

\textbf{Environments}

\begin{itemize}
  \item \texttt{abstract}, 44, 45
  \item \texttt{center}, 11, 12, 29, 87, 88, 148
  \item \texttt{description}, 74, 75
  \item \texttt{description*}, 75
  \item \texttt{document}, xxiii, 11, 12, 20, 41, 42
  \item \texttt{enumerate}, 73
  \item \texttt{enumerate*}, 75, 76
\end{itemize}

\textit{Formatting Information}
### Formatting Information

<table>
<thead>
<tr>
<th>Command</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>environment</td>
<td>xxx</td>
</tr>
<tr>
<td>equation</td>
<td>32</td>
</tr>
<tr>
<td>figure</td>
<td>89, 148</td>
</tr>
<tr>
<td>figure*</td>
<td>127</td>
</tr>
<tr>
<td>flushleft</td>
<td>88, 164</td>
</tr>
<tr>
<td>flushright</td>
<td>88</td>
</tr>
<tr>
<td>inparaenum</td>
<td>76</td>
</tr>
<tr>
<td>itemize</td>
<td>11, 73</td>
</tr>
<tr>
<td>itemize*</td>
<td>75</td>
</tr>
<tr>
<td>lrbox</td>
<td>101</td>
</tr>
<tr>
<td>minipage</td>
<td>100–102, 108, 168</td>
</tr>
<tr>
<td>multicol</td>
<td>127</td>
</tr>
<tr>
<td>picture</td>
<td>90, 91</td>
</tr>
<tr>
<td>quotation</td>
<td>xxx, 98, 148</td>
</tr>
<tr>
<td>raggedleft</td>
<td>29</td>
</tr>
<tr>
<td>raggedright</td>
<td>29</td>
</tr>
<tr>
<td>sbox</td>
<td>101, 102, 168</td>
</tr>
<tr>
<td>spacing</td>
<td>133</td>
</tr>
<tr>
<td>table</td>
<td>81, 82, 90, 148</td>
</tr>
<tr>
<td>table*</td>
<td>127</td>
</tr>
<tr>
<td>tabular</td>
<td>82, 88, 90, 101</td>
</tr>
<tr>
<td>Verbatim</td>
<td>105</td>
</tr>
<tr>
<td>verbatim</td>
<td>105</td>
</tr>
<tr>
<td>epsf</td>
<td>91</td>
</tr>
<tr>
<td>equation</td>
<td>32</td>
</tr>
<tr>
<td>error</td>
<td>messages, 243</td>
</tr>
<tr>
<td>Error messages</td>
<td>Capacity exceeded, 253</td>
</tr>
<tr>
<td></td>
<td>File not found, 254</td>
</tr>
<tr>
<td></td>
<td>Overfull hbox, 254</td>
</tr>
<tr>
<td></td>
<td>Runaway argument, 253</td>
</tr>
<tr>
<td></td>
<td>Too many )'s, 252</td>
</tr>
<tr>
<td></td>
<td>Undefined control sequence, 252</td>
</tr>
<tr>
<td></td>
<td>Underfull hbox, 253</td>
</tr>
<tr>
<td>etruscan</td>
<td>146</td>
</tr>
<tr>
<td>\EUR</td>
<td>17, 85</td>
</tr>
<tr>
<td>Euro</td>
<td>17</td>
</tr>
<tr>
<td>EuroMath</td>
<td>175</td>
</tr>
<tr>
<td>Evince</td>
<td>191, 222</td>
</tr>
<tr>
<td>ex</td>
<td>(relative measure), 25</td>
</tr>
<tr>
<td>example</td>
<td>78</td>
</tr>
<tr>
<td>exit</td>
<td>139, 246</td>
</tr>
<tr>
<td>extarticle</td>
<td>38</td>
</tr>
<tr>
<td>extreport</td>
<td>38</td>
</tr>
<tr>
<td>extsizes</td>
<td>38</td>
</tr>
<tr>
<td>fancybox</td>
<td>101, 102, 168</td>
</tr>
<tr>
<td>fancyhdr</td>
<td>135, 136</td>
</tr>
<tr>
<td>fancyhead</td>
<td>168</td>
</tr>
<tr>
<td>fancyvrb</td>
<td>105, 108</td>
</tr>
<tr>
<td>fbox</td>
<td>90, 101, 102, 156</td>
</tr>
<tr>
<td>fboxrule</td>
<td>101, 156</td>
</tr>
<tr>
<td>fboxsep</td>
<td>101, 156</td>
</tr>
<tr>
<td>fc-cache</td>
<td>139, 230</td>
</tr>
<tr>
<td>fc-cache</td>
<td>139</td>
</tr>
<tr>
<td>fc-list</td>
<td>230</td>
</tr>
<tr>
<td>fcolorbox</td>
<td>102, 156</td>
</tr>
<tr>
<td>figure</td>
<td>89, 148</td>
</tr>
<tr>
<td>figure*</td>
<td>127</td>
</tr>
<tr>
<td>figures</td>
<td>89</td>
</tr>
<tr>
<td>filenames</td>
<td>244</td>
</tr>
<tr>
<td>Finder</td>
<td>62, 157</td>
</tr>
<tr>
<td>fix-cm</td>
<td>152</td>
</tr>
<tr>
<td>float</td>
<td>80</td>
</tr>
<tr>
<td>floats</td>
<td>79, 89</td>
</tr>
<tr>
<td>flushleft</td>
<td>88, 164</td>
</tr>
<tr>
<td>flushright</td>
<td>148</td>
</tr>
<tr>
<td>flushleft</td>
<td>88, 164</td>
</tr>
<tr>
<td>flushright</td>
<td>88</td>
</tr>
<tr>
<td>fnpara</td>
<td>108</td>
</tr>
<tr>
<td>nsymbol</td>
<td>108</td>
</tr>
<tr>
<td>textbf</td>
<td>238</td>
</tr>
</tbody>
</table>

Formatting Information
textbf, 149
\textbf, 149
\fontenc, 19, 23
\fontencoding, 147
\fontfamily, 143, 148
\FontForge, 150, 232
\fontname, 234
\fontname, 233, 239
\fontfamily
\METAFONT, 137
changing temporarily, 146, 147
\colour, 155
\Computer Modern, 137
\encoding, 234
\families, 144
\in general, 137
\installing, 229
\METAFONT, 143
\PostScript, 143, 231
setting the default, 141
\sizes, 37
\TrueType, 143
\Type 1, 143
\fontsize, 152
\fontspec, 139, 141
\footcite, 118, 119
\footmisc, xxx
\footnote, xxx, 103, 107, 108
\footnote, 108
\footnotes, 107
\footnotesize, 152
\foreign, 154, 165
\foreignlanguage, 30
\FrameMaker, 174
\FreeType, 144
\gedit, 138
\geometry, 37, 58–60, 109, 131, 135
\Ghostscript, 189, 257, 259
\GIMP, 92, 96
gloss, 125
\gloss, 125
glossaries, 123
\glossary, 125
glossaries, 125, 126
glossary, 125
\GML, 3
\GNUPLOT, 96
graphics, 91
graphics, 61
\graphicspath, 96, 97
graphicx, 90, 91, 94
grep, 230
\textbf, 29, 147, 148
grouping, 148
groups, 148
\GSview, 189, 257–259
\H, 22
.h1, 3
:\h1, 3
\har2nat, 121
\textbf, 17
\headings, 135
cite, 135
\helvet, 143, 145
texbf, 151
\hline, 84
\hrule, 164
\hspace, 134
\HTML Tidy, 175
\huge, 152
\huge, 152
hyphen, 104
hyphenation, 26
\hyphenation, 26, 27
hyphens, 26
soft, 26

\i, 22
Illustrator, 96
ImageMagick, 93
images, 91
in (inches), 25
\includegraphics, 90–93, 96, 97, 103
InDesign, 174
\index, 123–125, 168
indexes, 123
InkScape, 96
inline, 75
inparaenum, 76
\input, 125
inputenc, 19, 21, 23, 32, 267
Installation
  Apple Mac, 189
  Linux, 189
  Mac OS X, 189
  Microsoft Windows, 189
  OS X, 189
  Unix, 189
Instant Preview, xxviii
\item, 72
itemize, 11, 73
itemize*, 75
\itshape, 149

JabRef, 113, 120, 263
Java, xxxi, 177
JSTOR, 113

kate, 138
kdvi, 191
Kile, xviii, xxvi, 6, 10, 222
kindlegen, 2
komascript, 35
kpdf, 191
kpsewhich, 228
kpsewhich, 158
Kword, 175

\L, 22
\l, 22
\label, 77, 81, 89, 110–112
\labelitemi, 171
\labelitemiv, 172
landscape, 86
\LARGE, 152
\Large, 12, 152
\large, 152
LaTeX, 179
latex, 8, 62, 193, 247, 249
latex-mode, 18
latex2rtf, 184
latexmk, 62, 125
latexmk, 247, 249
\leftmark, 137
textbf, 49

Lengths
  \baselinestretch, 133
  \columnsep, 127
  \fboxrule, 101, 156
  \fboxsep, 101, 156
  \parindent, 50, 101
  \parskip, 49–51
  \spaceskip, 26
  \tabcolsep, 86
  \textwidth, xxx

letter, 35
<table>
<thead>
<tr>
<th>Key</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>letterpaper</td>
<td>37, 38</td>
</tr>
<tr>
<td>letterspacing</td>
<td>134</td>
</tr>
<tr>
<td>linearb</td>
<td>146</td>
</tr>
<tr>
<td>\linebreak</td>
<td>166</td>
</tr>
<tr>
<td>Linux installation</td>
<td>189</td>
</tr>
<tr>
<td>Lisp</td>
<td>224</td>
</tr>
<tr>
<td>listings</td>
<td>104–106</td>
</tr>
<tr>
<td>\listoffigures</td>
<td>52</td>
</tr>
<tr>
<td>\listoftables</td>
<td>52</td>
</tr>
<tr>
<td>lists</td>
<td>72</td>
</tr>
<tr>
<td>bulletted</td>
<td>73</td>
</tr>
<tr>
<td>description</td>
<td>74</td>
</tr>
<tr>
<td>discussion</td>
<td>74</td>
</tr>
<tr>
<td>enumerated</td>
<td>73</td>
</tr>
<tr>
<td>inline</td>
<td>75</td>
</tr>
<tr>
<td>itemized</td>
<td>73</td>
</tr>
<tr>
<td>numbered</td>
<td>73</td>
</tr>
<tr>
<td>longtable</td>
<td>86</td>
</tr>
<tr>
<td>\lrbox</td>
<td>101</td>
</tr>
<tr>
<td>ls</td>
<td>227</td>
</tr>
<tr>
<td>\lstinline</td>
<td>104, 106</td>
</tr>
<tr>
<td>lwarp</td>
<td>183</td>
</tr>
<tr>
<td>lwarp, iv</td>
<td>183, 184</td>
</tr>
<tr>
<td>Mac OS X installation</td>
<td>189</td>
</tr>
<tr>
<td>macros</td>
<td>161</td>
</tr>
<tr>
<td>textbf</td>
<td>161</td>
</tr>
<tr>
<td>\makeatletter</td>
<td>163</td>
</tr>
<tr>
<td>\makeatother</td>
<td>163</td>
</tr>
<tr>
<td>makeglossaries</td>
<td>125</td>
</tr>
<tr>
<td>makeglossariesgui</td>
<td>125</td>
</tr>
<tr>
<td>\makeglossary</td>
<td>124</td>
</tr>
<tr>
<td>makeidx</td>
<td>123</td>
</tr>
<tr>
<td>makeindex</td>
<td>62, 123–125</td>
</tr>
<tr>
<td>\makeindex</td>
<td>123</td>
</tr>
<tr>
<td>\makeindex</td>
<td>124, 247, 249</td>
</tr>
<tr>
<td>\maketitle</td>
<td>42–45, 52, 58, 147, 162, 163, 253</td>
</tr>
<tr>
<td>man</td>
<td>124</td>
</tr>
<tr>
<td>Maple</td>
<td>96, 175</td>
</tr>
<tr>
<td>\marginal</td>
<td>109</td>
</tr>
<tr>
<td>marginal notes</td>
<td>109</td>
</tr>
<tr>
<td>margins</td>
<td>131</td>
</tr>
<tr>
<td>\markboth</td>
<td>135</td>
</tr>
<tr>
<td>\markright</td>
<td>135</td>
</tr>
<tr>
<td>textbf</td>
<td>2</td>
</tr>
<tr>
<td>marvosym</td>
<td>17, 84</td>
</tr>
<tr>
<td>math characters</td>
<td>31</td>
</tr>
<tr>
<td>MathCAD</td>
<td>95</td>
</tr>
<tr>
<td>Mathematica</td>
<td>96, 175</td>
</tr>
<tr>
<td>mathematics</td>
<td>xxiv, 31</td>
</tr>
<tr>
<td>MathJax</td>
<td>184</td>
</tr>
<tr>
<td>mathpazo</td>
<td>143, 145</td>
</tr>
<tr>
<td>mathptmx</td>
<td>143, 145</td>
</tr>
<tr>
<td>\mbox</td>
<td>27, 166</td>
</tr>
<tr>
<td>measurements</td>
<td>23</td>
</tr>
<tr>
<td>\medskip</td>
<td>132</td>
</tr>
<tr>
<td>memoir</td>
<td>xxx</td>
</tr>
<tr>
<td>memoir</td>
<td>35</td>
</tr>
<tr>
<td>Mendeley</td>
<td>113</td>
</tr>
<tr>
<td>textbf</td>
<td>15</td>
</tr>
<tr>
<td>textbf</td>
<td>42</td>
</tr>
<tr>
<td>Microbrew, see beer</td>
<td></td>
</tr>
<tr>
<td>Microsoft Windows installation</td>
<td>189</td>
</tr>
<tr>
<td>minipage</td>
<td>100–102, 108, 168</td>
</tr>
<tr>
<td>mirror</td>
<td>94</td>
</tr>
<tr>
<td>mkdir</td>
<td>217, 218, 231, 236</td>
</tr>
<tr>
<td>mktexlsr</td>
<td>65, 227</td>
</tr>
<tr>
<td>mm (millimeters)</td>
<td>25</td>
</tr>
<tr>
<td>textbf</td>
<td>248</td>
</tr>
<tr>
<td>multicols</td>
<td>127</td>
</tr>
<tr>
<td>\multicolumn</td>
<td>86, 88</td>
</tr>
</tbody>
</table>
textbf, 87
multirow, 86
mv, 248

My Computer, 62, 157, 206, 219
myheadings, 135

N800, xix
natbib, 118, 121
Nautilus, 157, 217
NetPBM, 184
newcent, 143, 145
\newcommand, 162, 167
\newcounter, 78
\newfontface, 146
\newfontfamily, 146
\newgeometry, 132
\newglossaryentry, 125
nimbus, 144
\noindent, 98
noitemsep, 74
normalem, 151
\normalsize, 152
nosep, 74
Notepad, xxii, 241
Notes, xxii

O, 22
\O, 22
\textbf, 17
\OE, 22
\oe, 22
Okular, 191, 222
oldgerm, 142
Omnimark, 176
\onehalfspacing, 133
oneside, 37
OpenOffice, xxii, 175, 177
OpenType, 144

Options
\p, 236
10pt, 37
11pt, 37
12pt, 11, 37
a4paper, 36, 38
b, 100
draft, 37
dvipsnames, 155
empty, 135
headings, 135
hyphen, 104
inline, 75
letterpaper, 37, 38
myheadings, 135
noitemsep, 74
normalem, 151
nosep, 74
oneside, 37
plain, 135
Scale, 141
scaled, 94, 145
Show Library Folder, 218
svgnames, 155
t, 100
T1, 23
titlepage, 37
twocolumn, 127
twoside, 37
unboxed, 75
utf8, 19
utf8x, 19, 23
V, 235

options, see Class Options
textbf, 57
Rogue, 124
OS X
installation, 189

Formatting Information
\ovalbox, 102
\oXygen, 177
\P, 110
package, xxx

Packages
  
  acronym, 126
  array, 85
  avant, 143
  babel, 20, 30, 50, 120
  bbding, 73, 171
  biblatex, ix, 5, 19, 112–114, 116–121
  biblatex-apa, 114
  biblatex-cheatsheet, 114
  biblatex-chicago, 114
  book, 47
  bookman, 143
  chancery, 143
  charter, 144
  colorbl, 86
  concrete, 142
  courier, 143
  csquotes, 120
  cypriot, 146
  datatool, 81, 86
  dcolumn, 86
  duerer, 142
  eiad, 142
  endnote, 108
  enumitem, 65, 74, 75, 77, 132
  epsf, 91
  etruscan, 146
  extarticle, 38
  extbook, 38
  extreport, 38
  extsizes, 38
  fancybox, 101, 102, 168
  fancyhdr, 135, 136
  fancyvrb, 105, 108
  fix-cm, 152
  float, 80
  fnpara, 108
  fontenc, 19, 23
  fontname, 233, 239
  fontspec, 139, 141
  footmisc, xxx
  geometry, 37, 58–60, 109, 131, 135
  gloss, 125
  glossaries, 125, 126
  glossary, 125
  graphics, 61
  graphicx, 90, 91, 94
  har2nat, 121
  helvet, 143, 145
  inputenc, 19, 21, 23, 32, 267
  komascript, 35
  landscape, 86
  linearb, 146
  listings, 104–106
  longtable, 86
  lwarp, iv, 183, 184
  makeidx, 123
  marvosym, 17, 84
  mathpazo, 143, 145
  mathptmx, 143, 145
  memoir, 35
  multicol, 127
  multirow, 86
  natbib, 118, 121
  newcent, 143, 145
  nimbus, 144
  oldgerm, 142
  package, xxx
  palatino, 11, 145
pandora, 142
paralist, 75–77, 255
parskip, 51
phoenician, 146
pifont, 73, 143
preview-latex, xxviii
pslatex, 145
ragged2e, 29
report, 47
rotating, 86
runic, 146
rustic, 142
section, 47, 132
sectsty, 47, 132
setspace, 133
soul, 29, 134
tabularx, 86
tabulary, 86
textcomp, 17, 18, 31, 73
times, 145
tocloft, 53
ulem, 151
uncial, 142
uni, 142
url, 11, 103, 104
urw, 144
utopia, 144
varioref, 111
verbatim, 105
xcolor, 57–59, 61, 102, 155, 156, 163, 311
packages, 56
documentation, 59
downloading, 59
indexing, 64
installing, 59, 61
textrb, 35, 56
page size

scaling, 259
PageMaker, 174
\pageref, 111
\pagestyle, 135, 136
PaintShop Pro, 96
palatino, 11, 145
Pandoc, 175, 182
pandora, 142
panels, 99
paper sizes, 36
\par, 88, 132, 148, 152, 164
\paragraph, 47, 77
paralist, 75–77, 255
\parbox, 99–101
\parencite, 117–119
\parindent, 50, 101
parskip, 51
\parskip, 49–51
\part, 47
\part*, 49
pc (picas), 25
PC\TeX, xxix
pdf2ps, 257
pdflatex, 8, 194
pdflatex, 247, 249
pdftotext, 186
person, 167, 168
textrb, 217
textrb, 217
phoenician, 146
Photoshop, 92, 96
picas, 25
picture, 90, 91
pifont, 73, 143
plain, 135
textrb, 1
points, 25

Formatting Information
PostScript, xviii, 63, 93, 97, 143, 144, 157, 158, 229, 231, 232, 234–236, 256, 257, 259
textbf, 20
Preview, xxv, 194, 222
preview, 255
preview-latex, xxviii
\printbibliography, 120
\printglossaries, 125
\printindex, 124
printing, 243, 258
reverse order, 259
selected pages, 259
\product, 154, 165
Products
AbiWord, 175, 177, 178
Acrobat Reader, 185, 194, 222, 258
afm2tfm, 234
apt, 190
apt-get, xxvi
Aquamacs, 222
ArchInfo, 96
AutoCAD, 95
BaKoMa \TeX, xxix
biber, ix, 5, 19, 62, 112–114, 120–122, 125
biblatex, 5, 112, 113
bibtex, 5, 62, 112–114, 118, 122, 125
Calibre, 2
cep, 93
charmap, 21
ChemDraw, 95
Chocolate Stout, 124
Command, 218
Computer, 157, 206, 219
configure, 65
Crayola, xv, 155
Cygwin, 67
DCF, 3
detex, 186
DocBook, 177
DocBook 5, xxxi
Dolphin, 62, 157, 217
Draw, 92, 96
dvialw, 260
dvieps, 260
dvihp, 260
dvip, 93, 97, 257, 259
dviview, 255
DynaTag, 176
El Capitan, 194
Emacs, xxvi, 10, 18, 86, 87, 222–224, 251, 259
emacs, 138
EuroMath, 175
Evince, 191, 222
fc-cache, 139, 230
Finder, 62, 157
FontForge, 150, 232
fontname, 234
FrameMaker, 174
FreeType, 144
gedit, 138
Ghostscript, 189, 257, 259
GIMP, 92, 96
GML, 3
GNUplot, 96
GSview, 189, 257–259
HTML Tidy, 175
Illustrator, 96
ImageMagick, 93
InDesign, 174
InKscape, 96
Instant Preview, xxviii
JabRef, 113, 120, 263
Java, xxxi, 177
JSTOR, 113
kate, 138
kdvi, 191
Kile, xviii, xxvi, 6, 10, 222
kindlegen, 2
kpdf, 191
kpsewhich, 228
Kword, 175
latex-mode, 18
latex2riff, 184
latexmk, 62, 125
Lisp, 224
lwarp, 183
makeglossaries, 125
makeglossariesgui, 125
makeindex, 62, 123–125
Maple, 96, 175
MathCAD, 95
Mathematica, 96, 175
MathJax, 184
Mendeley, 113
mktexlsr, 65, 227
My Computer, 62, 157, 206, 219
N800, xix
Nautilus, 157, 217
NetPBM, 184
Notepad, xxii, 241
Notes, xxii
Okular, 191, 222
Omnimark, 176
OpenOffice, xxii, 175, 177
OpenType, 144
oXygen, 177
PageMaker, 174
PaintShop Pro, 96
Pandoc, 175, 182
pdf2ps, 257
pdflatex, 8, 194
pdftotext, 186
PhotoShop, 92, 96
PostScript, xviii, 63, 93, 97, 143, 144, 157, 158, 229, 231, 232, 234–236, 256, 257, 259
Preview, xxv, 194, 222
ps2pdf, 257
psnup, 259
pstops, 259
Publisher, 174
PubMed, 113
qpdfview, 191, 222
Rexx, 224
rpm, xxvi
rtf2latex2e, 177, 184
Runoff, 3
Saxon, xxxi, 177, 180
Scientific Word, xxviii
Scribe, 3
Script, 3
Software Center, 190
Software Installer, xxvi
Synaptic, xxvi, 190, 191
t1binary, 232
t1utils, 232
\TeXShop, 20
\TeXshop, 194
tables-mode, 86
Terminal, 218
texconfig, 191, 193
texhash, 65, 227
\TeX Live, 191
texlive-full, 192
INDEX

\textbf{T}e\LaTeX\textspace, 119, 213, 215
\textit{TextEdit}, xxii
\textit{Textures}, xxviii
\textit{Thunar}, 62, 157, 217
\textit{Tidy}, 184
\textit{TikZ}, 91
\textit{tkPaint}, 96
\textit{tlmgr}, 57, 59, 65, 66
\textit{TrueType}, 144
\textit{Ubuntu}, 191
\textit{Ubuntu Software Centre}, 190
\textit{updmap}, 159
\textit{Velcro}, 165
\textit{Verilog}, 104
\textit{vi}, xxvi, 138, 223, 224
\textit{vptovf}, 234, 235
\textit{Web of Science}, 113
\textit{which}, 228
\textit{Windows}, xxv, 222
\textit{Windows 7/8}, 219
\textit{WinEdt}, xviii
\textit{Word}, xvi, xxii, xxiv, 18, 175–177, 183–185
\textit{WordPerfect}, xxii, 153
\textit{xdvi}, 255–258
\textit{XEdit}, 224
\textit{xkeycaps}, 21
\textit{XPress}, 174
\textit{XQuartz}, 255
\textit{yap}, 255
\textit{yum}, xxvi, 190
\textit{Zaurus}, xix
\textit{Zotero}, 113
\textit{textbf}, 248
\textit{\protect}, 108
\textit{\ProvidesPackage}, 238
\textit{ps2pdf}, 257
\textit{pslatex}, 145
\textit{psnp}, 259
\textit{pstops}, 259
\textit{pt (points)}, 25
\textit{Publisher}, 174
\textit{PubMed}, 113
\textit{qpdfview}, 191, 222
\textit{\quad}, 134
\textit{\quad}, 25, 134
\textit{quotation}, xxx, 98, 148
\textit{quotation marks}, 17
\textit{ragged2e}, 29
\textit{\RaggedLeft}, 29
\textit{\raggedleft}, 29, 85
\textit{\raggedright}, 29
\textit{\raggedright}, 29, 85, 100
\textit{\raggedright}, 29
\textit{\raisebox}, 171
\textit{\ref}, 77, 110–112
\textit{references}, 111
\textit{\reindex}, 167, 168
\textit{\textbf}, 248
\textit{\ren}, 248
\textit{\texbf}, 248
\textit{\renewcommand}, 45, 49, 87, 133, 163, 170
\textit{report}, 34
\textit{report}, 47
\textit{\RequirePackage}, 152
\textit{Rexx}, 224
\textit{\rightmark}, 137
\textit{\rm}, 248
\textit{\rmdefault}, 238
\textit{rotate}, 94
\textit{rotating}, 86
\textit{rpm}, xxvi

\textbf{Formatting Information}
FORMATTING INFORMATION

\textbf{rtf2latex2e}, 177, 184
RULE
\textit{em}, 28
\texttt{en}, 28
\texttt{\textbackslash rule}, 84
\texttt{runic}, 146
\texttt{Runoff}, 3
\texttt{rustic}, 142
\texttt{S}, 110
\texttt{Saxon}, xxxi, 177, 180
\texttt{Sbox}, 101, 102, 168
\texttt{Scale}, 141
\texttt{scale}, 94
\texttt{scaled}, 94, 145
\texttt{Scientific Word}, xxviii
\texttt{textbf}, 150
\texttt{Scribe}, 3
\texttt{Script}, 3
\texttt{\textbackslash scriptsize}, 152
\texttt{\textbackslash scshape}, 149
\texttt{secnumdepth}, 11, 49
\texttt{section}, 47, 132
\texttt{section}, 3, 47, 51, 170
\texttt{section}, xxx, 170, 171
\texttt{section numbering}, 48
\texttt{sections}, 47
\texttt{sectsty}, 47, 132
\texttt{\selectfont}, 147, 148, 150
\texttt{\selectlanguage}, 30
\texttt{\sentinel}, 167, 168
\texttt{\setcounter}, 49
\texttt{\setlength}, 49–51, 101
\texttt{\setmainfont}, 139
\texttt{\setmonofont}, 139
\texttt{\setsansfont}, 139
\texttt{setspace}, 133
\texttt{\sfdefault}, 238
\texttt{\sffamily}, 12, 149, 164
\texttt{\shadowbox}, 102, 168
\texttt{Show Library Folder}, 218
\texttt{sidebars}, 99
\texttt{\textbackslash singlespacing}, 133
\texttt{size (fonts)}, 151
\texttt{textbf}, 151
\texttt{\textbackslash slshape}, 149
\texttt{\textbackslash small}, 98, 152
\texttt{\textbackslash smallskip}, 133
\texttt{special characters}, 15, 31
\texttt{\textbackslash ss}, 22
\texttt{textbf}, 84
\texttt{style (fonts)}, 149
\texttt{\textbackslash subparagraph}, 47, 77
\texttt{\textbackslash subparagraph*}, 49
\texttt{\textbackslash subsection}, 47
\texttt{\textbackslash subsubsection}, 47
\texttt{sudo}, 138, 191
\texttt{summaries}, 44
\texttt{svgnames}, 155
\texttt{Synaptic}, xxvi, 190, 191
\texttt{t}, 100
\texttt{\textbackslash t}, 22
\texttt{T1}, 23
\texttt{t1binary}, 232
\texttt{t1utils}, 232
\texttt{T\textbackslash XShop}, 20
\texttt{T\textbackslash XShop}, 194
\texttt{\textbackslash tabcolsep}, 86
table, 81, 82, 90, 148
\textbf{table of contents}
  adding manual entry, 52
  automated entries, 51
\table*, 127
\tableofcontents, 14, 51, 52, \texttt{ trademark, 166}
\tablett, 150
  Textures, xxviii
\textwidth, xxx
\thechapter, 170
\theneumii, 78
\theneumii, 78
\theneumiv, 78
\thexample, 78
\TheSbox, 168
\thesection, 170
\textbf{thesis, 34}
\textwidth, 18, 19, 134
\thispagestyle, 135
\textbf{Thunar, 62, 157, 217}
\textbf{Tidy, 184}
\textbf{TikZ, 91}
textbf, 16
times, 145
\textbf{tiny, 152}
\textbf{title, 42, 44, 108, 162}
\textbf{titlecite, 118}
titlepage, 37
titles, 42
tkPaint, 96
tlmgr, 57, 59, 65, 66
\tmproduct, 165
tocdepth, 49, 52
tocloft, 53
tools, 107
tracking, see letterspacing
\TrueTipX, xxix
FORMATTING INFORMATION

\texttt{TrueType}, 144
\texttt{ttdefault}, 238
\texttt{ttfamily}, 149
\texttt{twocolumn}, 127
twoside, 37
typographics, 129
\texttt{u}, 22
Ubuntu, 191
Ubuntu Software Centre, 190
ulem, 151
\texttt{uline}, 151
unboxed, 75
uncial, 142
textbf, 151
uni, 142
units, 23
Unix
installation, 189
textbf, 148
\texttt{updmap}, 159
\texttt{upshape}, 149
url, 11, 103, 104
\texttt{url}, 104, 106, 108, 180
urw, 144
\texttt{usepackage}, 38, 57, 58, 91, 97, 106, 155, 159, 180, 231, 239, 255
\texttt{utf8}, 19
\texttt{utf8x}, 19, 23
utopia, 144
\texttt{V}, 235
\texttt{v}, 22
varioref, 111
Velcro, 165
\texttt{verb}, 102–106, 108
Verbatim, 105
verbatim, 105
verbatim, 105
verbatim text, 102
\texttt{VerbatimFootnotes}, 108
Verilog, 104
vi, xxvi, 138, 223, 224
viewer, 243
\texttt{vptovf}, 234, 235
\texttt{vspace}, 132, 134
\texttt{vspace*}, 133
\texttt{VTeX}, xxix

Web of Science, 113
which, 228
white-space
baselines, 133
double-spacing, 133
hard, 27
horizontal, 134
margins, 131
vertical
disappearing, 132
fixed, 132
flexible, 132
textbf, 14
Windows, xxv, 222
Windows 7/8, 219
WinEdt, xviii
Word, xvi, xxii, xxiv, 18, 175–177, 183–185
WordPerfect, xxii, 153
xcolor, 57–59, 61, 102, 155, 156, 163, 311
\texttt{xdvi}, 255–258
XEEdit, 224
\texttt{xelatex}, xxvii, 247, 249
\texttt{Xe\TeX}, xviii

Formatting Information
<table>
<thead>
<tr>
<th><strong>INDEX</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>xkeycaps, 21</td>
</tr>
<tr>
<td>XPRESS, 174</td>
</tr>
<tr>
<td>XQuartz, 255</td>
</tr>
<tr>
<td>yap, 255</td>
</tr>
<tr>
<td>yum, xxvi, 190</td>
</tr>
<tr>
<td>Zaurus, xix</td>
</tr>
<tr>
<td>Zotero, 113</td>
</tr>
</tbody>
</table>
Revision history

v7.81 — 19 November 2018  Updated and rewrote the section on image file formats

v7.8 — 25 February 2018  Updated BachoTeX and MarkupUP

v7.7 — 14 February 2018  Updated conference dates; added details of lwarp; fixed bibref pointer to PDF pages where given.

v7.6 — 14 May 2017  Added salient comments about how to choose an editor from Michael Sperberg-McQueen, and slides from TUG 2017 from Barbara Beeton on how to handle Mi\TeX errors.

v7.5 — 21 January 2017  Full update of Mac and Windows installation with new screenshots and a new layout for procedures; introduction of an XSLT routine to test when an element is immediately preceded by another element with only white-space intervening, and to generate a space token if needed (this overcomes the design flaw alluded to in the revision comments to v3.7 below); many updates to phraseology, and removal of obsolete mentions of packages and practices.

v7.45 — 4 November 2016  Updated details of TDS package installation

v7.44 — 11 October 2016  Updated details of meetings

v7.43 — 15 April 2016  Updated comments on superscripted ordinals, spacing around em rules, and details of loading OTF/TTF fonts

v7.42 — 1 March 2016  Edited all sections, corrected spellings, updated package details and all examples to Xe\TeX/biblatex/biber, fixed stray typos, checked links,. This is preparatory to v8, due for later in 2016.

v7.41 — 30 January 2016  Minor typos, updated event dates

v7.4 — 16 November 2015  Started updating converters

v7.3 — 26 October 2015  Updated meetings for 2016
Revision history

v7.2 — 12 July 2015 Several sections re-ordered to present the material is a more logical fashion. Numerous grammatical elisions corrected, and some late typos (thanks to Rob Borland).

v7.1 — 10 June 2015 Minor changes to accommodate revised PDF format.

v7.0 — 30 July 2014 Completely re-edited, large sections rewritten, obsolete material removed, including installation changes for the 2014 DVD, and a completely new responsive web site launched.

v6.0 — 30 December 2013 Updated links, replaced references to obsolescent packages, rewrote installation for the 2013 DVD

v5.7 — 21 December 2011 Moved and expanded the details of creating a personal TeX directory. Added new section on using the LaTeX Font Catalog.

v5.6 — 1 November 2011 Revised installation details for TL2011.

v5.5 — 25 May 2011 Minor revision; added details of page references in citations, a warning about the broken harvard.sty and its solution with natbib and har2nat, and a reference to bibunits.

v5.4 — 27 April 2011 Minor revision; added details of packages for body type size options; fixed bug in HTML bibrefs which were failing to retrieve the date.

v5.3 — 22 March 2011 Minor revision; removed mention of VTeX as a synchronous typographic editor and replaced with BaKoMa TeX. Located and fixed XSLT bug which was preventing cross-reference IDs being used correctly. Finally tracked down the non-appearance of italics in some places (no Lite Italic in my copy of Antique Olive).

v5.2 — 13 March 2011 Minor revision; even finer details of the problems Windows users face at installation.

v5.1 — 5 March 2011 Minor revision; spellings and font selection errors repaired; missing rule in HTML table example; better details of the problems Windows users face at installation.
Revision history

v5 — 28 January 2011 Major revision; Installation and Editors sections rewritten, remaining package references updated, and more new ones added.

v4 — 1 April 2009 Major revision; Installation and Editors sections reorganised, all package references updated, and new ones added.

v3.7 — 22 December 2006 There have again been some small but significant improvements, both in the \LaTeX code, and in the default installations and implementations. The default colour package is now \texttt{xcolor}; the default output for many people is now PDF; and the advent of \TeX means that TrueType fonts and Unicode are now much more easily supported. The DocBook DTD has been updated to 4.4, and the TypeBook DTD shim likewise, and the IGNOREd code from 3.5 and earlier versions has now finally been dropped. XSLT still has the notorious design flaw of ignoring whitespace nodes in mixed content when a DTD is used, but this seems to have gone unnoticed except by the publishing industry. The use of the \texttt{citetitle} element for bibliographic references has been replaced by \texttt{biblioref}.

v3.6 — 31 March 2005 Since the publication of the November 2003 edition in TUGboat, several new books on \LaTeX have been released, and this edition reflects some of the new material and approaches contained in them. See the Bibliography for details of these publications. The only technical change has been to use empty elements for the \TeX, \LaTeX, and other logos instead of the more usual entities so that the HTML version can use CSS to produce better logos. Thanks to whoever wrote the CSS for \TeX4ht, which is where I found the styles.

v3.5 — 29 July 2004 Modified DTD to add span element type to allow use of external entities for formatted \TeX, \LaTeX, and other logos in the HTML version. Changed entity declaration in the internal subset to enable this, and switched declarations and marked sections in the DTD. This now means it needs Saxon 7 or 8 to process, as Saxon 6 does not handle parameter entities values used as parameter entity declarations.
Revision history

v3.4 — 9 November 2003  Applied all Barbara Beeton’s corrections (see separate emails) and rewrote a few formatting macros to allow the document to fit more easily into US Letter shape. It would be nice if it would also format for A5 so that it could become a paperback but that’s another day’s work. Started on writing the missing sections (Installing Type 1 CM Fonts and Going beyond \LaTeX, but these are not finished yet) and rewrote entirely the existing (non-CM) Type 1 font installation procedure in line with the new (unreleased) Gutta-Percha script. Added hidden meanings for CD-ROM, DVD, IBM.

v3.3 — 20 August 2003  Fixed XSLT bug which wrongly lettered appendices. Fixed problem which called wrong font for examples of Times and Helvetica (thanks to William Adams). Updated numerous typos, added comments about pdftex option to color. Rewrote formatting for TUGboat.

v3.2 — 5 March 2003  Finished rewrite. Revised and expanded almost everything.

v3.1 — 28 August 2002  Recast in DocBook and reworded some sections. Started the big rewrite.