The University of York Department of Computer Science

Submitted in part fulfilment for the degree of MSc in Information Processing.

A guide to type-setting project reports in LaTeX 2_{ε} with the UoYCSproject class

Jeremy L. Jacob

Version 2.17, 2010-Nov-04

Supervisor: Jeremy L. Jacob

Number of words = 8832, as counted by `wc -w`.
This includes the body of the report, and Appendices **??**, **??** and **??**, but not **??**.

Abstract

LaTeX 2ε is a document markup and processing system built upon Donald Knuth's type-setting system, TeX.

UoYCSproject is a LaTeX 2ε class for producing reports describing projects taken as part of a taught course in the Department of Computer Science at the University of York. (It is not designed for research degree reports.)

A brief introduction to LaTeX 2ε is given. The UoYCSproject class is described.

This document itself is (inappropriately) an example of the use of the class UoYCSproject.

To all students everywhere

Acknowledgements

I would like to thank my goldfish for all the help it gave me writing this document.

As usual, my boss was an inspiring source of sagacious advice.

Contents

List of Figures

List of Tables

List of Listings

Part I

Preliminaries

1 Introduction

In each taught course, undergraduate or postgraduate, there is a compulsory large project.[1] By far the largest component of the assessment of the project is a written report. There are various appropriate technologies for producing reports. Among these is Lamport's LaTeX [**?**].

This user guide describes a LaTeX 2ε class, UoYCSproject, to help in the type-setting of project reports; it is (inappropriately) written using that document class. The division into parts, chapters and so on is too heavy for a brief introduction and user guide, but appropriate for a project report. The source code for this document is available through the CSW web site; you are welcome to use it as a template.

1.1 What is LaTeX?

LaTeX, or more strictly, LaTeX 2ε, is a notation for describing document structure (much as HTML or XML applications) [**?**]. It is very different from WYSIWYG, which has been characterised as "What you see is all you've got"[2]

LaTeX 2ε is built on top of Donald Knuth's TeX [**?**]. TeX is a notation for describing type-set pages *plus* a macro language. LaTeX 2ε is a collection of TeX macros that allows for extensions and modifications using the class and package mechanisms. Thus a LaTeX 2ε description of a document can be turned into print by processing it with a suitable program.

Output is available as the original Device Independent (DVI) format (by using latex to process the document), PostScript (by converting from DVI) or PDF (by using pdflatex to process the document).

TeX itself was developed by Donald Knuth for type-setting his books, particularly his multi-part work on algorithms [**? ? ?**]; take a look at

[1]Except for the three teaching-year joint degrees with mathematics, where a computer science project is optional.

[2]**?** , p7, Footnote 1] says that "Brian Reid attributed this phrase to himself and/or Brian Kernighan".

them to see what is possible. He also developed a font design program to accompany TeX, METAFONT.[3]

1.2 Advantages of TeX

TeX has a very sophisticated text type-setting algorithm; its implementation is proved optimal (Donald Knuth did more or less found the theory of algorithms). The PDFTeX engine extends the algorithm to include hanging punctuation, for even better results. (See the TeX showcase for several examples; it lives at http://www.tug.org/texshowcase/.)

TeX has a very sophisticated mathematics type-setting algorithm.

TeX also has a Turing-equivalent macro language so that you can program substructures in your document.

1.3 Advantages of LaTeX 2ε

LaTeX 2ε provides a pre-defined set of document structures (using the TeX macro language), and hooks for integrating further structures.

LaTeX 2ε simplifies the task of writing TeX macros (unless you need something very sophisticated).

1.4 Advantages of a programmable mark-up language

I consider the ability to write definitions the greatest advantage of TeX-like systems.

Such a facility enables its users to design a collection of macros that reflect the abstract syntax of important structures in the document (later we will see an example of part of a collection of macros for describing cryptographic protocols). Just doing this will help you ask the right questions about your project, even if you end up using some other document processing system. The fact that LaTeX 2ε also lets you associate type-setting commands with each element of the abstract syntax is an

[3] An illustration of how Donald Knuth's mind works. The current version of TeX is 3.141592; the next version, should there be one, will be numbered 3.1415926, and the one after that 3.14159265. On his death the source code is to be amended to print out 'Version \Pi', and no further changes will be allowed. Similarly, METAFONT version numbers are converging on *e*; currently it is Version 2.71828.

added bonus, and one that gives you consistent type-setting across the document, and between documents.

2 Useful references

2.1 Books

?] The original source. It has a reasonable reference manual, but can be terse. It does not cover package and class writing, nor does it cover more than a handful of useful packages. It does describe the BibTeX and index making programs.

?] A comprehensive reference; it covers everything except the many add-on packages. Most people use this as their primary reference.

?] A guide to many of the most useful add-on packages and classes.

?] A slightly dated guide to packages for graphics.

?] A slightly dated guide to packages for adding hyperlinks, and producing PDF and HTML from LaTeX 2_ε.

2.2 Papers

There are many papers describing LaTeX 2_ε and its associated packages. They are available on-line, usually through the Comprehensive TeX Archive Network.[1]. They are usually also available on the TeX Live distribution,[2] which the department uses.[3]

The useful *general* papers are:

The Not So Short Introduction to LaTeX 2_ε [?]

Available in several languages.

Be warned that this paper describes the standard classes. There are a few differences in the class options and declarations bewteen the standard classes and UoYCSProject.

[1]CTAN, http://www.ctan.org/.

[2]http://www.tug.org/texlive/

[3]Departmental Linux users should look under file:///usr/local/pkg/ for the current TeX Live distribution, and under that for the various doc directories; documentation is usually in pdf or dvi files.

Math mode [?] A detailed explanation of typesetting mathematics in LaTeX 2ε.

The Comprehensive LaTeX Symbol List [?] An enormous list of symbols and how to make them.[4]

Packages in the 'graphics' bundle [?] A bit out of date (it does not describe PDF extensions), but a useful introduction.

Hypertext marks in LaTeX [?] Access to hypertext features via the hyperref package.

Most of the effects happen automatically on loading the package.

It works best in combination with the hypcap package.

UoYCSproject loads these packages for you, and sets some of the manual things to sensible values.

The KOMA-Script bundle [?]

UoYCSproject is based on the KOMA-Script scrreprt class. The manual will tell you about several extra facilities available to you (but you should not change layout, and such things).

2.3 Web resources

The Comprehensive TeX Archive Network <http://www.ctan.org/>

What it says on the label. Almost everything you need in the way of TeX and friends can be found here. Also known as CTAN.

The TeX Users Group (TUG) <http://www.tug.org/> A useful web site.

TUG members get the TeX Live distribution as part of their subscription.

TeX FAQ <http://faq.tug.org/>

An extremely useful first port of call for solving common problems, hosted by TUG.

The PracTeX Journal <http://tug.org/pracjourn/>

An on-line journal of TeX practice, including a Q&A section.

[4]An experimental web application for finding symbols can be found at http://detexify.kirelabs.org/.

The LaTeX Project <http://www.latex-project.org/>
The centre of the LaTeX project.

The TeX newsgroup <news:comp.text.tex> If asked politely, questions not in the FAQ or standard sources of documentation will usually be answered by gurus. *Minimal* examples of problems, together with the versions of TeX, LaTeX and all classes and packages used in the example must be given.

Peter Flynn's 'Formatting information' <http://research.silmaril.ie/latex/>
An on-line LaTeX 2_ε manual.

The beauty of LaTeX <http://dartar.free.fr/w/?wakka=latex> A page describing typographic advantages of TeX-based systems over common competitors.

A wiki for LaTeX 2_ε <http://en.wikibooks.org/wiki/LaTeX> A relatively new resource; as good or as bad as a Wiki can be.

A Visual FAQ for LaTeX 2_ε <http://www.tex.ac.uk/tex-archive/info/visualFAQ/visualFAQ.pdf>
The associated README file for this resource says:

> Having trouble finding the answer to a LaTeX question? The Visual LaTeX FAQ is an innovative new search interface that presents over a hundred typeset samples of frequently requested document formatting. Simply click on a hyperlinked piece of text and the Visual LaTeX FAQ will send your Web browser to the appropriate page in the UK TeX FAQ.

MathTran instant preview <http://www.mathtran.org/toys/jfine/editor2.html>
A web-based application to let you try out small pieces of TeX (*not* LaTeX 2_ε) source code (especially mathematical source code) to see what the type-set version looks like.

3 The LaTeX edit-process cycle

The standard books on LaTeX describe the process by which you turn your document description into ink. Most describe this process using latex, which produces DVI format, and a DVI viewer, such as xdvi.

Since these books were written it has become more convenient to use pdflatex, which produces PDF format, and a PDF viewer such as xpdf or acroread (xpdf is slightly more convenient than acroread, although it does not support all the features that acroread does, nor does it have as good rendering).

LaTeX 2ε source may be created using any editor. Several editors have support for TeX and LaTeX 2ε, including managing the edit-create cycle. I like emacs with the AUCTeX enhancements to the TeX modes; Windows users often use WinEDT.

The perfect edit-process cycle goes like this:

1. Create a LaTeX 2ε source file, and any others needed, such as a BibTeX file, figures, and so on.

2. Run pdflatex. (This creates PDF output with place-holders for missing information and auxiliary files with information about the table of contents, cross references, name of file(s) containing the bibliographic database, and so on.)

3. Run BibTeX. (This creates a file containing the references.)

4. Run pdflatex. (This recreates PDF output with place-holders for missing information and auxiliary files with information about the table of contents, cross references, name of file(s) containing the bibliographic database, and so on, but this time also with bibliographic citations.)

5. Run pdflatex. (This will create PDF output which is complete.)

Imperfections in this cycle creep in when you make errors in the files, add new citations, and so on. Further recompilation is necessary; rerunning

BibTeX is only necessary if new citations are inserted or if an entry in the bibliographic database changes.

Tools such as AUCTeX/ emacs and WinEDT can manage the process for you.

A brief guide to using LaTeX 2_ε on (some of) the department's systems is given in **??**.

Part II

Concepts of LaTeX 2_ε

In this part of the document I briefly review some of the main concepts of LaTeX 2_ε documents.
This is *not* a comprehensive guide to LaTeX 2_ε, but a list of useful concepts, together with a few hints and tips. Consult the main references for full details.

4 The anatomy of a LaTeX 2$_\varepsilon$ source file

The layout of a normal LaTeX 2$_\varepsilon$ document description is given in **??**.

On Line 1 is the *document class declaration*. This declares the class to which the document belongs, as the mandatory parameter to the documentclass command; mandatory parameters appear in curly braces. Most classes have optional parameters; these are passed in the square brackets. Optional parameters for most commands appear in an unusual position when they do appear: between the command name and the mandatory parameters.

Next (represented by Line 2 of **??**) is the preamble, which contains further definitions and declarations for the document. This can stretch over many lines. Usually there is a great deal of freedom about what can appear here; the class UoYCSproject is very restricted, and introduces a separate mechanism for private declarations (see **??**).

The document body is delimited by the markers on Line 3 and Line 5. In between goes the document structured into (optional parts,) chapters, sections, subsections and so on, represented here by Line 4.

```
1  \documentclass[class options]{class name}
2    preamble (definitions and declarations)
3  \begin{document} % this is a comment, from the ‘%’ to the ‘<cr>’.
4    \maketitle % to generate the title information
5    body
6  \end{document}
```

Listing 4.1: The anatomy of a LaTeX 2$_\varepsilon$ file

5 Definitions and Declarations

5.1 Declarations

Declarations are easiest to deal with, so we describe them first. There are two kinds: individual items and packages of related items.

5.1.1 Individual declarations

Most classes and packages allow or mandate features of the document to be set by declaration. The syntax is a command that names the declaration and a parameter that gives the value. For example, all classes that have a title have a declaration to set it: see **??**. Along with the title usually goes an author (or authors) and an optional date (if not given, the date defaults to the date the file is processed); again see **??**.

Some classes, such as UoYCSproject, have a larger collection of declarations. (The declarations made available by UoYCSproject are given in **??**.)

5.1.2 Package loading

Often a document contains structures that are orthogonal to the document structure. A common example in computer science projects is a code listing. A *package* is a collection of definitions that supports marking up the structures. The listings package is recommended for marking up code fragments (that package has been used for the fragments of LaTeX 2ε code in this document).

```
\title{text}
\author{name 1 \and name 2 \and name 3}
\date{text}
```

Listing 5.1: Declaring title matter

```
\usepackage{listings} % for pretty printed code listings
```

Listing 5.2: Loading a package

```
\newcommand*{\uoy}{The University of York}
```

Listing 5.3: A new command without parameters

Note that UoYCSproject provides a different, non-standard, place for you to load packages. See **??**.

Packages are loaded with the command **usepackage**{package name}. An example is given in **??** They often have large numbers of optional parameters, and associated declarations to control their behaviour. The description should be given in the package documentation.

There are very many packages available; see examples given in **??** and **??** and the web site http://www.tex.ac.uk/tex-archive/help/Catalogue/.

5.2 Definitions

It is the ability to make definitions that gives LaTeX 2ε its real power. Commands can be defined to express the logical structure of the concepts in your project, and these can be separated from their mark-up.

Note that UoYCSproject provides a different, non-standard, place for you to load packages. See **??**.

There are two kinds of definitions: commands and environments.

5.2.1 Commands

New commands are declared with the **newcommand** or **newcommand***
command.

The simplest use is when you have a long phrase that you need to type regularly, and you wish to save yourself some keystrokes and/or ensure consistency between occurrences. An example is given in **??**. Anywhere that '\uoy' occurs in the scope of the definition the text 'The University of York' is substituted. The definition is designed to be used in a *text mode* rather than a *math mode* (see **??**).[1]

[1]If called in a math mode the result is '$TheUniversityofYork$'!

```
\newcommand*{\msg}[3]{#1\rightarrow#2:#3}
```

Listing 5.4: A new command with parameters

```
\newcommand*{\msg}[3]{%
  #2\Longleftarrow\left[#3\right]\Longleftarrow#1}
```

Listing 5.5: A second new command with parameters

Commands can also have parameters; and this is where the two forms of definition differ from each other. \newcommand* defines a command whose parameters may *not* include paragraph breaks ('short' parameters in TEX parlance); \newcommand defines a command whose parameters *may* include paragraph breaks ('long' parameters in TEX parlance). The 'starred' form is almost always the appropriate one.

As an example, **??** shows how to define a command, called \msg, to typeset a message in a protocol; the message has three parts: sender, intended recipient and body. The command is to be used in a math mode, and later we define an environment for whole protocols.

The optional parameter following the name of the command being defined is the number of parameters (maximum: 9) that the command has; these parameters are called #1, #2 and #3. As an example of its use, '$A \rightarrow B : M, K(A, B, N)$' may be typeset by the call '\msg{A}{B}{M,K(A,B,N)}'.

Now suppose that you wish to change the printed format of a message everywhere in the document: all you need to do is to modify the body of the definition. Alternative definitions are given in **??** and **??**. The second

```
\newcommand*{\msg}[3]{
  \begin{array}{@{}c@{}}
       #1
    \\ \bigtriangledown
    \\ #3
    \\ \bigtriangledown
    \\ #2
  \end{array}
}
```

Listing 5.6: A third new command with parameters

```
\begin{itemize}
\item The first bullet point.
\item And now the second.
\item Followed by a third.
\end{itemize}
```

Listing 5.7: An example of a bulleted list

definition of \msg typesets the call '\msg{A}{B}{M,K(A,B,N)}' as

$$B \Longleftarrow [M, K(A, B, N)] \Longleftarrow A$$

while the third typesets it as

$$\begin{array}{c} A \\ \bigtriangledown \\ M, K(A, B, N) \\ \bigtriangledown \\ B \end{array}$$

The **newcommand** commands will report an error if the command name is already defined (possibly in the environment). You can *redefine* a command by using **renewcommand** and **renewcommand***. There are other subtle variations on command definition, including the ability define commands with one optional parameter. For these, see a standard book on LaTeX 2ε, such as those listed in **??**.[2]

5.2.2 Environments

An *environment* is used to group together a structure. An instance of environment e begins with **begin**{e} and ends with **end**{e}.

For example, there are predefined environments for various types of lists (see **??**), for quotations (see **??**; note the use of a comment to break a long word and hide the new-line character and the use of \- to state additional places where hyphenation is allowed; this example is typeset in **??**), and for arranging formulæ (the array environment in **??**).

Declaring an environment is very like making a definition, except that now we have to give code for the start and the end of the environment.

[2]The underlying TeX definition mechanism is extremely powerful, allowing a much greater flexibility in the syntax of introduced commands. See **?**].

```
As \citet[P~1 complete]{Joyce:FW} most eloquently says:
\begin{quotation}\small
  riverrun, past Eve and Adam's, from swerve of shore to bend of
  bay, brings us by a commodius vicus of recirculation back to Howth
  Castle and Environs.

  Sir Tristram, violer d'amores, fr'over the short sea, had
  passencore rearrived from North Armorica on this side the scraggy
  isthmus of Europe Minor to wielderfight his penisolate war: nor
  had topsawyer's rocks by the stream Oconee exaggerated themselse
  to Laurens County's gorgios while they went doublin their mumper
  all the time: nor avoice from afire bellowsed mishe mishe to
  tauftauf thuartpeatrick: not yet, though venissoon after, had a
  kidscad buttended a bland old isaac: not yet, though all's fair in
  vanessy, were sosie sesthers wroth with twone nathandjoe. Rot a
  peck of pa's malt had Jhem or Shen brewed by arclight and rory end
  to the regginbrow was to be seen ringsome on the aquaface.

  The fall
  (baba\-badal\-gharagh\-takammi\-narronn\-konn\-bronn\-% break long word
  ton\-ne\-rronn\-tuonn\-thunn\-trovar\-rhoun\-awn\-% break long word
  skawn\-too\-hoo\-hoor\-den\-ent\-hur\-nuk!)\ of
  a once wallstrait oldparr is retaled early in bed and later on
  life down through all christian minstrelsy. The great fall of the
  offwall entailed at such short notice the pftjschute of Finnegan,
  erse solid man, that the humptyhillhead of humself prumptly sends
  an unquiring one well to the west in quest of his tumptytumtoes:
  and their upturnpikepointandplace is at the knock out in the park
  where oranges have been laid to rust upon the green since
  devlinsfirst loved livvy.
\end{quotation}
and then later, in the last lines of the book, \citep[P~627,
Lines 8---16]{Joyce:FW}:
\begin{quote}\small
  Yes. Carry me along, taddy, like you done through the toy fair! If
  I seen him bearing down on me now under whitespread wings like
  he'd come from Arkangels, I sink I'd die down over his feet,
  humbly dumbly, only to washup. Yes, tid. There's where. First. We
  pass through grass behush the bush to. Whish! A gull. Gulls. Far
  calls. Coming, far! End here. Us then. Finn, again! Take.
  Bussoftlhee, mememormee! Till thousendsthee. Lps. The keys to.
  Given! A way a lone a last a loved a long the
\end{quote}
```

Listing 5.8: An example of quotations

```
\newenvironment*{mq}[1]
  {\begin{quote}\small\itshape\newcommand*{\cl}{#1}}% begin code
  {\par\hspace*{\fill}---\citep{\cl}\end{quote}}% end code
```

Listing 5.9: A new quote environment

In **??** I show how to define an environment that behaves like the quote environment, except that the font used is small and italic. It also takes one parameter, a citation label, which causes the citation to be printed at the bottom right hand side of the quote, preceded by an em-dash. The command used to define the environment is \newenvironment* (there is also an un-starred version, as well as 'renew' versions). The environment's name is mq. It has one parameter. Next comes the code to be executed at the start of the environment: begin a quote environment, set the font size to small and its shape to italic, and finally store the parameter value in the macro definition \cl (the parameter is only accessible as #1 in the begin code). Last comes the code executed at the end of the environment: force a paragraph break, produce just enough white space so that the citation is right-justified, an em-dash and then the citation itself.

As a second example, consider the \msg command, to typeset one message in a protocol. The protocol itself is best captured as a list of messages. To do this we define an environment, yet another version of \msg and a 'and then do' command; these are given in **??**.

Note that the definitions of the commands are made local to the environment and cannot be accessed outside it (the counter declaration must, alas, be global). Because the definition of \msg is nested one level deep its parameters have names that start with *two* hashes, ##1 and so on.

An example of their use is given in **??** (This protocol is due to **?**]; the notation $\{M\}_K$, marked up as \enc{K}{M}, means the encryption of M under symmetric key K). The typeset version is Protocol **??** on Page **??**.

```
\newcounter{msgnumber}
\newenvironment*{protocol}
{ % begin code
  \setcounter{msgnumber}{0}%
  \newcommand*{\msg}[3]{%
    \refstepcounter{msgnumber}\themsgnumber&##1&##2&##3}
  \newcommand*{\next}{\\}
  \begin{math}\displaystyle%
    \begin{array}{r@{.\quad}l@{\rightarrow}l@{\;:\;}l}%
}
{ % end code
    \end{array}%
  \end{math}%
}
```

Listing 5.10: An environment and a command to typeset protocols

```
\begin{protocol}
  \msg{A}{B}{M,A,B,\enc{K_{AS}}{N_{A},M,A,B}}
  \next
  \msg{B}{S}{M,A,B,\enc{K_{AS}}{N_{A},M,A,B},%
                  \enc{K_{BS}}{N_{B},M,A,B}}
  \next
  \msg{S}{B}{M,\enc{K_{AS}}{N_{A},K_{AB}},%
              \enc{K_{BS}}{N_{B},K_{AB}}}
  \next
  \msg{B}{A}{M,\enc{K_{AS}}{N_{A},K_{AB}}}
\end{protocol}
```

Listing 5.11: Markup for the Otway-Rees protocol

6 The body of the document

6.1 The anatomy of the body

The body of the document has a structure given in **??**.

There are usually three parts to a report:

Front matter The title page, dedication, acknowledgements, abstract, tables of contents and so on.

Most of this is taken care of automatically by the UoYCSproject class (as long as you provide the declarations). However, there are some optional features of the document (such as figures and tables) whose use cannot be detected. If you do use them you should indicate this by asking for the appropriate lists to be included.

Main matter The content of the document, appropriately structured.

In UoYCSproject the document is structured into chapters, with, optionally, a coarser structuring into parts (other classes have other rules). The chapters can be structured into sections, the sections into subsections, and so on, using the commands given in **??**. You should not miss out a level of headings. Note that \paragraph and \subparagraph are historical names that refer to titled sectional units, not to a coherent collection of sentences; a sectional (sub-)paragraph may well be composed of several coherent collections of sentences.

Sections are numbered, and copied to the table of contents, as low as subsections. (If you really want to change the depth of the table of contents you can, although it is *deprecated*, by altering the value of the tocdepth counter. For example, **setcounter**{tocdepth}{3} would cause sub-subsections to be numbered. In the UoYCSproject class you should do this in the local definitions file.)

Sometimes a title will be too long for the table of contents or the running headings. A shorter, optional, title can be given to the command; the short title is used instead of the long one in both the table of contents and the running headings. For example, see **??**.

```
% FRONT MATTER
\listoffigures % Optional. Generates a list of figures in the document.
\listoftables % Optional. Generates a list of tables in the document.
% Optional. Other list−generating commands specific to your document.
% (For example, the listings package has a command
% \lstlistoflistings to produce a list of code listings.)
% MAIN MATTER
\part{title} % Repeat as often as necessary, perhaps zero times.
\chapter{title} % Repeat as often as necessary, but at least once.
\section{title} % Repeat as often as necessary, perhaps zero times.
\subsection{title}% Repeat as often as necessary, perhaps zero times.
\subsubsection{title}% Repeat as often as necessary, perhaps zero times.
\paragraph{title}% Repeat as often as necessary, perhaps zero times.
\subparagraph{title}% Repeat as often as necessary, perhaps zero times.
% BACK MATTER
\bibliography{file1,file2} % Construct bibliography from databases in
                                        % 'file1.bib' and 'file2.bib'.
\appendix % remaining chapters to be numbered as appendices
\chapter{title} % Repeat as often as necessary, perhaps zero times.
\section{title} % Repeat as often as necessary, perhaps zero times.
\subsection{title} % Repeat as often as necessary, perhaps zero times.
\subsubsection{title} % Repeat as often as necessary, perhaps zero times.
\paragraph{title} % Repeat as often as necessary, perhaps zero times.
\subparagraph{title} % Repeat as often as necessary, perhaps zero times.
```

Listing 6.1: The anatomy of the body in UoYCSproject

```
\chapter[The truth]{An accurate, complete and verisimilitudinous %
  account of the happenings that occurred at that time and place}
```

Listing 6.2: A sectional unit with an optional short title

Back matter The references and appendices.

> Appendices are just chapters, although they will be numbered differently.
>
> There are various means of producing a bibliography or list of references. The best way is through BibTEX, a format for bibliographic databases that is integrated with LATEX 2ε.

Not mentioned in **??** are other parts of documents usually found in the front or back matter, such as glossaries and an index. LATEX 2ε has facilities to produce both of these. Only a glossary is worth including in a project report, and is usually small enough to be done by hand as an appendix. A good index is very hard to produce, and not worth the trouble for a project report (until you turn it into a book, that is!).

6.2 Splitting the document up

Sometimes it is convenient to break a document into pieces. LATEX 2ε provides two mechanisms for doing this.

The command **input**{<file>} searches for a file called '<file>.tex' and includes it. The effect is as if the file was typed in place.

The command **include**{<file>} searches for a file called '<file>.tex' and includes it. The file should contain a complete chapter, and must start a new page. The **includeonly** command can be used to selectively process chapters, speeding up processing time in the drafting phase. Page ranges and labels from the last run of missing chapters are taken account of by this mechanism, so a small edit to one chapter may mean only reprocessing that chapter. See the standard documentation.

6.3 Text elements

6.3.1 Modes

LATEX 2ε text is processed in various *modes*. The same input will give different results in each mode. The modes include:

paragraph for ordinary text,

left-to-right for text that will not be broken across lines,

Here is some text incorrectly placed in math mode — note how different it is from paragraph mode:
$Hereissometextincorrectlyplacedinmathmode - - - notehowdifferentitisfromparagraphmode.$

Figure 6.1: The result of treating text as mathematics

math for mathematics (actually there are two variants, in-line and displayed), and

picture for drawing simple pictures.

It is rare to be caught out by the wrong mode, as LaTeX 2ε usually switches automatically when necessary, and most of the time you can forget about modes. The most common mistake is to use a command in a text mode that only makes sense in math mode, when LaTeX 2ε will report an error. The reverse mistake —to place text in a math mode— results in ugly output; see **??**.

6.3.2 Simple paragraphs

A paragraph (in the sense of a coherent collection of sentences and not in the sense of a sectional unit) is just a block of text. Paragraphs are separated by blank lines (that is, sequences of at least two new line characters). Words in a paragraph are separated by sequences of spaces and at most one newline. See **??**, where the first quotation consists of three paragraphs.

Where necessary a paragraph break can be forced by a **par** command. The indentation on the first line of a paragraph can be suppressed by beginning the paragraph with a **noindent** command.

6.3.3 Characters

Reserved characters

There are some characters which are reserved and may not be used in text. These are listed in **??**, together with how to make them if you really need them.

#	$	%	&	~	_	^	\	{	}
\#	\$	\%	\&	\textasciitilde	_	\^{}	\textbackslash	\{	\}

Table 6.1: Reserved characters and how to make them. Note that the two braces are only defined in math mode.

Name	Character	Mark-up	Mode	Comment
Hyphen	-	-	Text	To join two words, as in 'Kraft-Ebbing'.
en-dash	–	--	Text	To form a range, as in 'The period 1997–2003'.
em-dash	—	---	Text	To separate two phrases — or use as parenthesis brackets.
Minus sign	$-$	$-$	Math	To indicate subtraction, as in '$2003 - 1997$'.

Table 6.2: Dashes and their use

Ellipses

Sometimes you will need to show that words have been left out of a quotation. This is done by a mark called an *ellipsis* '. . .'; it can be made by the 'low dots' command **ldots**. The output of **ldots** is not the same as three full stops: compare a. . . to...z. It is bad style to let a sentence trail off with an ellipsis. . .

For mathematics, centred dots (**cdots**) look better: $1 + \frac{1}{2} + \cdots + \frac{1}{2^n} + \cdots + \frac{1}{256}$. (Some people think that $\sum_{n=0}^{8} \frac{1}{2^n}$ looks even better.)

A vertical ellipsis can be made with **vdots**; an example of its use can be seen in **??**.

Dashes

Another class of characters that sometimes causes confusion are the various dashes. See **??**.

Many, many special characters and symbols are available. Some are available automatically, some are parts of packages that you will need to load explicitly. See 'The Comprehensive LaTeX Symbol List'.

Spaces

Spaces are a special case (in this section space characters are typeset thus: '␣'). There are three kinds:

1. '␣' An ordinary space (or a non-empty sequence of spaces). May be printed as an inter-word space, an inter-sentence space or a newline. Under certain circumstances (for example, in a math mode or immediately following a command) it may be ignored.

   Most of the time you should use ordinary spaces to separate items.

2. '~' A *tie*. It will never be replaced by anything other than an inter-word space.

   Ties should be used whenever you want to suppress a line-break. In particular, they should be used in constructs such as `Mr~Smith`, `Hypothesis~C`, and so on.

3. '\␣' A hard space. It may be replaced by an inter-word space or a newline. It may not be ignored.

   Hard spaces are useful when you want to force a new line, and TeX does not think it is building a line: use `\␣\newline`.

   A hard space is used to protect an inter-word space immediately following a command name. For example '`\TeX␣is␣useful`' typesets as 'TeXis useful', while '`\TeX\␣is␣useful`' typesets as 'TeX is useful'. (An alternative method is to place an empty pair of braces after the command, for example '`\TeX{}␣is␣useful`'; this has the advantage that it works no matter what the following character, for example '`\TeX{}:␣useful!`'.)

   Their other use is to prevent LaTeX 2ε from thinking it is at a sentence end when it is not. LaTeX 2ε (because TeX does) treats a space as an inter-sentence space if it is preceded by a non-uppercase character and a full-stop. This can happen with abbreviations, e. g. 'etc.'. (Compare the spaces after the 'e.' and the 'g.' with the inter-word and inter-sentence space on the same line.)

Series	\mdseries	Medium Series
	\bfseries	**Boldface Series**
Family	\rmfamily	Roman Family
	\sffamily	San Serif Family
	\ttfamily	`Typewriter Family`
Shape	\upshape	Upright Shape
	\itshape	*Italic Shape*
	\slshape	*Slanted Shape*
	\scshape	SMALL CAPS SHAPE

Table 6.3: Font attribute declarations

\tiny	\scriptsize	\footnotesize	\small
abcXYZ	abcXYZ	abcXYZ	abcXYZ
\normalsize	\large	\Large	\LARGE
abcXYZ	abcXYZ	abcXYZ	abcXYZ
\huge		\Huge	
abcXYZ		abcXYZ	

Table 6.4: Font size declarations

Character attributes

It is possible to vary the series, shape, family, size and colour of *text* fonts (see the references for the attributes for mathematical fonts). This is *deprecated* in the text, but recommended in the implementation of abstract syntax. See **??**. There is also a \normalfont declaration when all else fails.

To each font declaration there is a command, of the form \\textXX{text}, where the XX should be replaced by the first two letters of the corresponding declaration: for example, '\textsc{text}' produces 'TEXT'. The exception to the rule is '\textnormal{text}'.

Font size is controlled by the declarations given in **??**. (Not all font sizes may be available; if not available something close will be chosen.)

To change colours you need to load the color package. You get a declaration, \\color{colour} and its associated command \\textcolor{colour}{text}. You also get coloured backgrounds and framed boxes. A few colours (red, blue, green, cyan, yellow, magenta, black, white) are pre-defined;

you must define others yourself. *The use of colour is deprecated: if you must use it, do so very, very carefully.*

6.3.4 Emphasised text

Emphasised text should be marked up logically, using **emph**. The command is context dependent and can be nested. For example,[1]

```
\textnormal{Sometimes \emph{we \emph{discover} unpleasant} truths.}
```

typesets as

> Sometimes *we* discover *unpleasant* truths.

Use of **textit**{...} or **itshape** to simulate the same effect is deprecated.

6.3.5 Lists

LaTeX 2ε has three kinds of lists available:

bulleted made with the itemize environment,

numbered made with the enumerate environment, and

labelled made with the description environment (this list is an example of the description environment).

Each has a similar format. Individual items are introduced by **item**; in the case of the description environment the **item** command has an 'optional' parameter for the label (which *must* be present). LaTeX 2ε changes the numbering and bulleting styles for sub-lists, to a reasonable depth (if you exceed this it probably means you have a poorly structured document). See **??**.

It is possible to define your own list structures, and this is a common way of building an abstract syntax for a document-specific structure (for example, a variant of the enumerated list would have been a good way to build a protocol display). See a standard reference (**??**).

[1]The first sentence of EWD498, but with my emphasis!

```
\begin{description}
\item[Thing One] likes
  \begin{enumerate}
  \item Green Eggs and
  \item Ham
  \end{enumerate}
\item[Thing Two] has never seen either
  \begin{itemize}
  \item a Star−Bellied Sneetch or
  \item a Lorax.
  \end{itemize}
\end{description}
```

Listing 6.3: Examples of lists

6.3.6 Quotations

In-line quotations

A running quotation in text must be surrounded by quote marks. There are two kinds, double and single. Opening single quotes are made with the “ `” character, and the corresponding close quote is made with the “' ” character. Double quotes are made with *pairs* of single quotes: “``” and “''”; the double-quote character “"” is never used.[2]

Displayed quotations

There are two kinds of displayed quotation:

1. the quote environment, and
2. the quotation environment.

The two environments are very similar. The quote environment is recommended for short quotes and the quotation environment for long quotes. Both are illustrated in **??**.

[2]Actually, it is used. One use is in the sentence to which this footnote is attached. Another use is in code listings for programming languages that, for example, use the character to delimit strings. It is also the name of the command that produces the ‘¨’ accent in words such as ‘coördinate’; accents are not discussed in this document.

WARNINGS

BEWARE ⇒

- Neither of the displayed quotation environments adds quotation marks.

  Check the Student Handbook to find out if we currently require quotation marks around displayed quotes. If they are required you will need to add them manually.

BEWARE ⇒

- Departmental rules require a citation with all quotes. These must be supplied manually. See **??** and **??** for examples.

6.3.7 Bibliographies

Various packages have been written to enhance the presentation of bibliographies and citations. The UoYCSproject class loads the natbib style and sets up citations to follow the Departmental approved style (IEEE). The UoYCSproject class also fixes the bibliography style for the approved departmental style.

Lists of references can be generated from a database in BIBTEX format. This is a flat text file. The documentation for the IEEEtran BIBTEX styles [?] will tell you how to format this file. The references for this document are an example. Each entry has the following layout:

```
@Entry_Type{Label,
 Field0   =   {Value0},
 Field1   =   {Value1},
              ⋮
 Fieldn   =   {Valuen}
}
```

There are many different entry types and each type has a different array of compulsory and optional fields. There are two features of BIBTEX that cause problems when preparing the bibliographic database: see **??**.

Citations are of two types, parenthesized and textual. If Joyce:FW is a label associated with the record for James Joyce's *Finnegans Wake* then

Parenthesized	\citep{Joyce:FW}	generates [?]
	\citep[\S4]{Joyce:FW}	generates [? , §4]
	\citep[see][\S4]{Joyce:FW}	generates [see ? , §4]

Textual	\citet{Joyce:FW}	generates ?]
	\citet[\S4]{Joyce:FW}	generates ? , §4]
	\citet[see][\S4]{Joyce:FW}	generates ? , §4]

If several citations are applicable they can be included in the same citation command, as a comma separated list, without spaces; for example: '. . . important modernist works~\citep{Elliot:WL,Joyce:FW}' might produce '. . . important modernist works [17, 18]'.

The natbib package provides several other facilities for typesetting parts of citations, such as titles; see its documentation.

6.3.8 Floats

A *float* is a numbered item, usually with a caption, that can 'float' around the document and gets a special entry in the front matter. In this document there are three classes of float, *tables* (for example, **??**), *figures* (for example, **??**) and *listings* (for example, **??**). The contents of the item need have no relation to the class of float, although it is helpful to the reader if they do!

Control of floats and their positioning is a complex subject, and apart from mimicking examples in the source of this document you really ought to consult a standard reference (see **??**).

Each float can have a symbolic label for use by LaTeX 2ε's cross-referencing mechanism.

6.3.9 Tabulating data

The common thing to find in a table float is a tabular environment. These are very flexible, and too complex to describe in this note. Simple examples may be seen in **??** and **??**. There are also packages to give tabular environments with extra functionality. See a standard reference (**??**).

6.3.10 Theorem-like environments

LaTeX 2ε allows you to define special series of named and numbered paragraphs, called theorem-like environments. The canonical examples are theorems, lemmata and hypotheses. Theorem-like environments have an optional parameter for textually naming the content of the environment, in addition to numbering it.

In **??** I define one for numbering protocols. My usual habit is to combine a theorem-like environment with an ordinary one to get a 'close-paragraph' marker; and I have done that here. (A better solution would

```
\newtheorem{PROTOCOL}{Protocol}
\newenvironment{prot}[1][]
{\newcommand*{\tmp}{#1}
  \ifthenelse{\equal{\tmp}{\empty}}
     {\begin{PROTOCOL}}
     {\begin{PROTOCOL}[\tmp]\ \newline}
}
{\newline\hspace*{\fill}
  \rule{0.666666em}{1.07867788em} % Golden ratio (approx)
\end{PROTOCOL}}
```

Listing 6.4: A theorem-like environment for protocols

be to roll the protocol and prot environments together; I have separated them here for illustration.) As an example here is **??**, typeset using

```
\begin{prot}[Otway−Rees]. . . \end{prot}.
```

Protocol 1 (Otway-Rees)

1. $A \rightarrow B$: $M, A, B, \{N_A, M, A, B\}_{K_{AS}}$
2. $B \rightarrow S$: $M, A, B, \{N_A, M, A, B\}_{K_{AS}}, \{N_B, M, A, B\}_{K_{BS}}$
3. $S \rightarrow B$: $M, \{N_A, K_{AB}\}_{K_{AS}}, \{N_B, K_{AB}\}_{K_{BS}}$
4. $B \rightarrow A$: $M, \{N_A, K_{AB}\}_{K_{AS}}$

∎

6.3.11 Mathematics

LaTeX 2ε has superb type-setting facilities for mathematics (see **??**). For complex work the AMSTeX packages (developed by the American Mathematical Society) will handle everything you could possibly need. Most people only need the basic, pre-loaded LaTeX 2ε facilities.[3]

In-line mathematics can be produced using the ‘math-shift’ construction, \$. . . \$. For example, \$A+B\$ produces ‘$A + B$’. An alternative is to use the math environment.

Displayed mathematics is made using the displaymath, equation, eqnarray and eqnarray* environments, depending on the exact effect desired.

[3]There is a school of thought that mistakes were made in the basic, pre-loaded LaTeX 2ε facilities, particularly the eqnarray environment. The solution proposed by this school is to always load and use the AMSTeX packages.

$$\int_{-1}^{1} \frac{(T_n(x))^2}{\sqrt{1-x^2}}\,dx = \begin{cases} \pi & \text{if } n = 0 \\ \pi/2 & \text{if } n \in \mathbb{N}_1 \end{cases}$$

Figure 6.2: An example of mathematical type-setting. (Othogonality in Chebyshev polynomials; T_n is the nth Chebyshev polynomial: $T_n(x) = \cos(n \cos^{-1} x)$.)

```
\section[Brief titles]{How to avoid tedious prolixity in the titles
  of sections, when they are printed in the Table of Contents}
\label{sec:brief}
```

Listing 6.5: An example of a label

The subject is too complex to discuss here, and the standard references (**??**) should be consulted.

6.3.12 Cross references

LaTeX 2ε has a powerful cross-reference mechanism. Anything for which a number can be generated (parts, chapters, sections, tables, figures, theorem-like structures, equations and so on) can have a symbolic label. See **??**, where a section is given the label sec:brief.

The section number can be referred to by using '**ref**{sec:brief}'. You can also refer to the page on which the section occurs by using the command '**pageref**{sec:brief}'. (See the examples in **??**.)

The hyperref package (loaded as part of UoYCSproject) automatically turns references made with **ref** and **pageref** into internal links when a suitable format is output (for example, PDF). It also adds three further commands:

```
In Section~\ref{sec:brief}, starting on Page~\pageref{sec:brief}, we
see how to do it.

Sections~\ref{sec:long}--\ref{sec:brief} report this in detail.
```

Listing 6.6: Examples of cross-references

- \**ref***, which does *not* make the internal hyperlink.
- \\autoref, which (sometimes) adds the name of the type of unit; for example the command \\autoref{sec:brief} will generate 'section 3.2' (or what ever number it turns out to be). The hyperref package makes the whole phrase into an internal link.

  (If \\autoref causes problems the easiest thing to do is fall back on \ref and await the bug fixes.)
- \\nameref, which typesets the name of the section.

6.3.13 Pictures

Pictures in LaTeX 2_ε can either be

- drawn within LaTeX 2_ε's native picture environment (if they are simple)
- drawn by a more sophisticated package, such as pgf/tikz and pdftricks, or
- imported using an external format, using a package such as graphics (you need to be careful with formats; pdflatex cannot accept PostScript, Encapsulated or otherwise; PNG or PDF works best[4]).

See the documentation given in **??**.

[4]There is a Linux program to convert from Encapsulated PostScript to PDF, epstopdf.

Part III

The document class UoYCSproject

7 The document class UoYCSproject

7.1 The antecedents of UoYCSproject

The LaTeX 2ε class UoYCSproject is based on the KOMA-Script class scrreprt and so has most of the facilities provided by that class. However some, such as page layout and the title declarations, are fixed or redefined. (For the record, the following options are passed to scrreprt: fontsize=11pt, a4paper, twoside[1], abstracton, numbers=noenddot, BCOR13mm, DIV=calc.)

The ifthen package is provided.

UoYCSproject chooses the font encoding (T1) using the fontenc package and font sets by packages from the PSNFSS bundle [**?**] (for roman shape: Hermann Zapf's Palatino, the University's font, using the mathpazo package; for san serif shape: Helvetica, using the helvet package with a scaling of 0.9; for typewriter shape: Courier, using the courier package), while accessing the micro-typographic features of pdfetex (character protusion and font expansion) via the microtype package.

British English hyphenation and names are set, using the babel package. (If you need them, babel allows you to include other languages in your document.)

The bibliography and citation styles are fixed using natbib, setting the bibliography style to IEEEtranN.

Hyperlinks are produced using the hyperref package. This package also produces bookmarks and sets some of the PDF 'document properties'. Anchor placement in floats is improved by loading the hypcap package, with parameter all.

The UoYCSproject class works with the versions available as part of the TeX-Live 2007 distribution (http://www.tug.org/texlive/).

7.2 Declarations for the title pages

The available declarations are listed in **??**.

[1]You must print a document of class UoYCSProjct double-sided.

Declaration	Parameter	Optionality
title	{short text}	C
author	{short text}	C
date	{short text}	O
\abstract	{long text}	C
\wordcount	{short text}	C
\includes	{short text}	O
\excludes	{short text}	O
\dedication	{short text}	O
\acknowledgements	{long text}	O
\BEng	—	1
\BSc	—	1
\MEng	—	1
\MMath	—	1
\SWE	—	1
\SCSE	—	1
\MIT	—	1
\MNC	—	1
\GTC	—	1

Table 7.1: Declarations of class UoYCSproject.
The declarations typeset in **bold, san-serif font** are common to many classes; the remainder are peculiar to UoYCSproject. Where declarations take parameters the type of the parameter, short (paragraph breaks forbidden) or long (paragraph breaks allowed) is given.
The optionality tags have the following meanings: 'C': compulsory; 'O': optional; '1': choose exactly one of this group.

The **title**, **author** and **date** declarations are standard. You should use them to record: the *title of your report*, *your name* and the *date of submission* respectively. If the date is omitted a message giving the date of processing is produced; this should not be on your final submission!

You are required to produce an abstract. Most classes achieve this by an abstract environment in the body (including the KOMA-Script classes). This is changed by UoYCSproject to an \abstract declaration in the preamble.

You also need to give the word count of the parts of the document to be marked. There is a compulsory declaration, \wordcount, to state the actual word count of the main body of the report.[2] Optionally you can generate text that states which extra sections are included, and which excluded by the \includes and \excludes declarations. If both optional declarations are omitted the message produced is:

> "This includes the body of the report only."

If the inclusions only are given, the message produced is:

> "This includes the body of the report, and <include text>."

If the exclusions only are given, the message produced is:

> "This includes the body of the report, but not <exclude text>."

If both are given the message produced is:

> "This includes the body of the report, and <include text>, but not <exclude text>."

You should also state which qualification the project contributes to by using exactly one of the declarations: \BEng, \BSc, \MEng, \MMath, \SWE, \MIT, \GTC or \SCSE (\MIP is available for historical purposes!).
These take no parameter.

You may generate a page with a dedication and/or acknowledgements on it by using the declarations \dedication and \acknowledgements.

Users of the **include** mechanism may add an includeonly declaration.

The title pages are typeset in the usual way, by a **maketitle** command as the first command in the body of the document.

[2] Under Un*x you can do this by running wc −w on the file. If you split the document between files you can use a command of the pattern cat file1 file2 file3 | wc −w. An alternative is to use the TeXcount utility (see http://tug.ctan.org/pkg/texcount) which has a web interface at http://folk.uio.no/einarro/Services/texcount.html.

7.3 Loading your own packages and adding your own commands

Because UoYCSproject needs to carefully control the order of package loading you should include nothing in the preamble other than the declarations given in **??**.

A non-standard mechanism is provided for loading your own packages and declaring your own commands and environments. If your main file is called <main>.tex, then the extra preamble should go in a file called <main>.ldf (for *L*ocal *Def*initions).

7.4 Other non-standard facilities

7.4.1 Citations

The citation mechanism in UoYCSproject is different from the standard, which uses the command **cite**. It uses the more flexible scheme implemented by the natbib package, of \citep for parenthesised citations and \citet for citations as text. See **??**.

The command **cite** is defined to be the same as \citet (which is probably not what you want).

7.4.2 Cross references

The standard mechanism (**ref**{label}) works, but \autoref{label} is preferred. The \autoref command generates the location type (section, subsection, or whatever) as well as the location number. See **??**.

Part IV

Appendices

A Packages not pre-loaded that you may find useful

These are packages that might help you with special tasks in writing your report. (I have omitted specialist packages that some people might find useful, such as the package which provides support for Braille.)

If these packages are not in our standard TEX-Live release they can be obtained from the Common TEX Archive Network (CTAN); this is easiest via the catalogue (http://www.tex.ac.uk/tex-archive/help/Catalogue/). Even if we do have the packages, you may wish to check for later versions.

A.1 Main document

array Improves the facilities for tabular and array environments.

acronym Helps you manage acronyms, ensuring that all are printed in full at least once. It can generate a list of used acronyms, too.

amsmath Enhanced mathematical type-setting; there are several ancillary packages.

calc Allows easier arithmetic calculations than native mode. The documentation comes with a syntax, formal semantics and implementation scheme, as well as an informal narrative.

changebar Allows you to indicate changes to your document by a bar in the margin. (Useful for showing drafts to your supervisor.)

glossaries To aid production of a glossary.

graphics To include pictures from external sources

graphicx Like 'graphics', but with a 'key=value' interface.

listings To pretty-print code listings. Several languages are predefined, and you can define your own.

movie15 To insert moving images in the PDF. Particularly useful for presentations of physical artefacts (see **??**).

pdfcomment Allows you to take advantage of the PDF comment and annotation facilities, for on-line copies but *not* the printed copy. (PDF comments are not widely supported outside of Adobe products.)

pdfpages Allows you to include a PDF document inside your LaTeX 2_ε document. It is very flexible, allowing you to select pages, print *n* logical pages per physical page, and so on.

pgf/tikz Allows more complex drawings than native LaTeX 2_ε mode. Suitable for all flavours of LaTeX 2_ε. There is a very well designed font end to pgf, called TikZ, that makes drawing diagrams much easier.

siunitx For consistent typesetting of physical quantities.

todonotes Allows you to insert 'to do' markers that are visually obvious, and to generate a table-of-contents-like list of them.

A.2 Presentations

You may need to give a presentation, and there are several LaTeX 2_ε packages for preparing slides; a good list may be found at http://www.miwie.org/presentations/.

The main advantage of using a LaTeX 2_ε-based solution for presentations is being able to re-use your LaTeX 2_ε source and avoid re-typing everything for PowerPoint (or similar presentation tool, such as the one in OpenOffice). The package of choice for most people is beamer. Section 5 of the Beamer *User's Guide* gives a lot of good advice on creating presentations, even (or especially) if you opt to use PowerPoint.

B Common LaTeX 2ε 'Gotchas'

There are just a few things that trip up a newcomer to TeX, LaTeX 2ε and BibTeX.

B.1 Parameterless macros gobble white space

B.1.1 Problem

Any macro gobbles all the white space following up to the next non-white space character. This does not matter if the next thing is a parameter, but it does matter otherwise. For example, '`\LaTeXe␣is␣easy.`' typesets as 'LaTeX 2εis easy.'.

B.1.2 Solution

A solution is to always protect the white space by preceding it with a backslash: '`\LaTeXe\␣is␣easy.`'. Another solution is to always follow a parameterless macro with empty braces: '`\LaTeXe{}␣is␣easy.`'. Both typeset as 'LaTeX 2ε is easy.'. The second solution is more robust if the white space is replaced by something else, such as punctuation.

B.2 Confusion between end of abbreviation and end-of sentence

B.2.1 Problem

TeX treats a full stop, '.' between a non-capital letter and white space as indicating an end of a sentence, and so it generates a sentence-separating space rather than a word-separating space. The problem most commonly arises with abbreviations such as 'etc.', 'i. e.', 'e. g.' and so on.[1]

[1]Some people think it is better style to use 'and so on', 'that is' and 'for example', and so on, neatly avoiding the problem; it also avoids the quite common confusion between 'i. e.' and 'e. g.'.

B.2.2 Solution

Protect any such spaces with a backslash. \␣ is always treated as an inter-word space. See **??**.

B.3 Wrong type of dash

There are four different types of dash available in TeX-based systems. The different dashes have different uses: see **??** for a discussion.

B.4 Wrong type of quote

In TeX-based systems quotation marks come in balanced pairs:

- '6-9 quote marks' (enlarged: ' ... ')
- "66-99 quote marks" (enlarged: " ... ")

None of these are made using the '"' key. The '6' quote marks are produced by a different key to the '9' quote marks. See **??**.

B.5 'Fragile' commands in 'moving' arguments

B.5.1 Problem

Some arguments to LaTeX 2ε commands are known as *moving arguments*. These are arguments that are potentially typeset elsewhere in the document (and may or may not be typeset at the point they occur). An example is the title of a chapter; as well as being typeset at the point it occurs it will be typeset a second time as part of the table of contents.

Some LaTeX 2ε commands are *fragile*: they break if moved. These are few in number, and they are rarely used in moving arguments (this document contains none). Examples include **footnote**{...}, **begin**, **end** and all commands with an optional argument.

```
\chapter{Typesetting footnotes\footnote{Footnotes are fragile}}
```

causes an error.

B.5.2 Solution

Fragile commands in moving arguments should be *protected*.

```
\chapter{Typesetting footnotes\protect\footnote{Footnotes are fragile}}
```

does not break.

That example would give an odd-looking table of contents. Even better is:

```
\chapter[Typesetting footnotes]{%
  Typesetting footnotes\footnote{Footnotes are fragile}}
```

which moves the optional argument and typesets the compulsory argument in place. (See also **??**.)

B.6 BibTEX gotchas

There are two features that often catch people out when preparing bibliography files.

1. Multiple authors in an author field must be separated with 'and':

   ```
   author = {John Smith and Brown, Mary and Joe Green and Lillian White}
   ```

   Commas must *not* be used to separate names as they are used to indicate a surname occurring before the forename (as in 'Brown, Mary').

2. BibTEX may change capitalisation of your text. Capital letters that must stay as capital letters should be protected from BibTEX's formatting by braces:

   ```
   title = {A Guide to {C++}: Its use with {Z}, {B} and {Alloy}}
   ```

C Running LaTeX on the departmental systems

C.1 Under GNU/Linux

The default departmental TeX-and-friends installation is the latest TeXLive distribution (see http://www.cs.york.ac.uk/support/texlive.php). You will need /usr/local/bin in your PATH to access it.

You may also wish to set a variable called TEXINPUTS if you have private collections of macros. This variable is a colon-separated list of directories (just like PATH) for TeX to search. If you keep everything in the same directory then the default value should be good enough. The default value is .::, the empty directory name meaning 'the standard library installed with TeXLive'.

To run LaTeX 2_ε from the command line on a file called foo.tex you type the command 'pdflatex foo' to produce a PDF file called foo.pdf. Similarly, to extract the bibliographic information associated with foo.tex you should run 'bibtex foo' *after* running LaTeX 2_ε — see **??**.

If you want to use AUCTeX with emacs then you need to put

```
(load "auctex.el" nil t t)
(load "preview-latex.el" nil t t)
```

in your .emacs file.

C.2 Under Microsoft Windows

I have never done this. I am told that WinEDT is the tool to use.

D ?? typeset

The typeset version of **??** is between the horizontal lines. Note that TEX could not find an ideal break for the 3rd paragraph (unsurprisingly, as this is a difficult text to typeset), and so has let one line protrude too far.

As **?** , P 1 complete] most eloquently says:

> riverrun, past Eve and Adam's, from swerve of shore to bend of bay, brings us by a commodius vicus of recirculation back to Howth Castle and Environs.
>
> Sir Tristram, violer d'amores, fr'over the short sea, had passencore rearrived from North Armorica on this side the scraggy isthmus of Europe Minor to wielderfight his penisolate war: nor had topsawyer's rocks by the stream Oconee exaggerated themselse to Laurens County's gorgios while they went doublin their mumper all the time: nor avoice from afire bellowsed mishe mishe to tauftauf thuartpeatrick: not yet, though venissoon after, had a kidscad buttended a bland old isaac: not yet, though all's fair in vanessy, were sosie sesthers wroth with twone nathandjoe. Rot a peck of pa's malt had Jhem or Shen brewed by arclight and rory end to the regginbrow was to be seen ringsome on the aquaface.
>
> The fall (bababadalgharaghtakamminarronnkonnbronntonnerronntuonnthunntrovarrhounawnskawntoohoohoordenenthurnuk!) of a once wallstrait oldparr is retaled early in bed and later on life down through all christian minstrelsy. The great fall of the offwall entailed at such short notice the pftjschute of Finnegan, erse solid man, that the humptyhillhead of humself prumptly sends an unquiring one well to the west in quest of his tumptytumtoes: and their upturnpikepointandplace is at the knock out in the park where oranges have been laid to rust upon the green since devlinsfirst loved livvy.

and then later, in the last lines of the book, [**?** , P 627, Lines 8–16]:

> Yes. Carry me along, taddy, like you done through the toy fair! If I seen him bearing down on me now under whitespread wings like he'd come from Arkangels, I sink I'd die down over his feet, humbly dumbly, only to washup. Yes, tid. There's where. First. We pass through grass behush the bush to. Whish! A gull. Gulls. Far calls.

Coming, far! End here. Us then. Finn, again! Take. Bussoftlhee, mememormee! Till thousendsthee. Lps. The keys to. Given! A way a lone a last a loved a long the