

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 $\text{\LaTeX} 2_{\epsilon}$ with the UoYCSproject class

Jeremy L. Jacob

Version 2.18, 2012-May-29

Supervisor: Jeremy L. Jacob

Number of words = 8832, as counted by `wc -w`.
This includes the body of the report, and Appendices A, B and C,
but not Appendix D.

Abstract

$\text{\LaTeX } 2_{\epsilon}$ is a document markup and processing system built upon Donald Knuth's type-setting system, \TeX .

UoYCSproject is a $\text{\LaTeX } 2_{\epsilon}$ 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 $\text{\LaTeX } 2_{\epsilon}$ 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

I	Preliminaries	13
1	Introduction	15
1.1	What is \LaTeX ?	15
1.2	Advantages of \TeX	16
1.3	Advantages of $\text{\LaTeX} 2_{\epsilon}$	16
1.4	Advantages of a programmable mark-up language	16
2	Useful references	19
2.1	Books	19
2.2	Papers	19
2.3	Web resources	20
2.4	Video	22
3	The \LaTeX edit-process cycle	23
II	Concepts of $\text{\LaTeX} 2_{\epsilon}$	25
4	The anatomy of a $\text{\LaTeX} 2_{\epsilon}$ source file	27
5	Definitions and Declarations	29
5.1	Declarations	29
5.1.1	Individual declarations	29
5.1.2	Package loading	29
5.2	Definitions	30
5.2.1	Commands	30
5.2.2	Environments	32
6	The body of the document	37
6.1	The anatomy of the body	37
6.2	Splitting the document up	39

Contents

6.3	Text elements	40
6.3.1	Modes	40
6.3.2	Simple paragraphs	40
6.3.3	Characters	41
6.3.4	Emphasised text	44
6.3.5	Lists	44
6.3.6	Quotations	45
6.3.7	Bibliographies	46
6.3.8	Floats	47
6.3.9	Tabulating data	47
6.3.10	Theorem-like environments	47
6.3.11	Mathematics	48
6.3.12	Cross references	49
6.3.13	Pictures	50
III	The document class UoYCSproject	51
7	The document class UoYCSproject	53
7.1	The antecedents of UoYCSproject	53
7.2	Declarations for the title pages	53
7.3	Loading your own packages and adding your own commands	56
7.4	Other non-standard facilities	56
7.4.1	Citations	56
7.4.2	Cross references	56
IV	Appendices	59
A	Packages not pre-loaded that you may find useful	61
A.1	Main document	61
A.2	Presentations	62
B	Common \LaTeX2ϵ ‘Gotchas’	63
B.1	Parameterless macros gobble white space	63
B.1.1	Problem	63
B.1.2	Solution	63
B.2	Confusion between end of abbreviation and end-of sentence	63
B.2.1	Problem	63

B.2.2	Solution	64
B.3	Wrong type of dash	64
B.4	Wrong type of quote	64
B.5	'Fragile' commands in 'moving' arguments	64
B.5.1	Problem	64
B.5.2	Solution	65
B.6	BIBTEX gotchas	65
C	Running L^AT_EX on the departmental systems	67
C.1	Under GNU/Linux	67
C.2	Under Microsoft Windows	67
D	Listing 5.8 typeset	69

List of Figures

6.1	The result of treating text as mathematics	40
6.2	An example of mathematical type-setting	49

List of Tables

6.1	Reserved characters and how to make them	41
6.2	Dashes and their use	42
6.3	Font attribute declarations	43
6.4	Font size declarations	44
7.1	Declarations of class UoYCSproject	54

List of Listings

4.1	The anatomy of a $\text{\LaTeX} 2_{\epsilon}$ file	27
5.1	Declaring title matter	29
5.2	Loading a package	30
5.3	A new command without parameters	30
5.4	A new command with parameters	31
5.5	A second new command with parameters	31
5.6	A third new command with parameters	32
5.7	An example of a bulleted list	32
5.8	An example of quotations	33
5.9	A new quote environment	34
5.10	An environment and a command to typeset protocols . . .	35
5.11	Markup for the Otway-Rees protocol	35
6.1	The anatomy of the body in <code>UoYCSproject</code>	38
6.2	A sectional unit with an optional short title	39
6.3	Examples of lists	45
6.4	A theorem-like environment for protocols	48
6.5	An example of a label	49
6.6	Examples of cross-references	50

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 [1].

This user guide describes a $\text{\LaTeX}_{2\epsilon}$ 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, $\text{\LaTeX}_{2\epsilon}$, is a notation for describing document structure (much as HTML or XML applications) [1]. It is very different from WYSIWYG, which has been characterised as “What you see is all you’ve got”²

$\text{\LaTeX}_{2\epsilon}$ is built on top of Donald Knuth's \TeX [2]. \TeX is a notation for describing type-set pages *plus* a macro language. $\text{\LaTeX}_{2\epsilon}$ is a collection of \TeX macros that allows for extensions and modifications using the class and package mechanisms. Thus a $\text{\LaTeX}_{2\epsilon}$ 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 [3–5]; take a look at them

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

²Lamport [1, p7, Footnote 1] says that “Brian Reid attributed this phrase to himself and/or Brian Kernighan”.

to see what is possible. He also developed a font design program to accompany T_EX, METAFONT.³

1.2 Advantages of T_EX

T_EX 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 PDF_TE_X engine extends the algorithm to include hanging punctuation, for even better results. (See the T_EX showcase for several examples; it lives at <http://www.tug.org/texshowcase/>.)

T_EX has a very sophisticated mathematics type-setting algorithm.

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

1.3 Advantages of L^AT_EX 2_ε

L^AT_EX 2_ε provides a pre-defined set of document structures (using the T_EX macro language), and hooks for integrating further structures.

L^AT_EX 2_ε simplifies the task of writing T_EX 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 T_EX-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 L^AT_EX 2_ε also lets you associate type-setting commands with each element of the abstract syntax is an

³An illustration of how Donald Knuth's mind works. The current version of T_EX 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.

1.4 Advantages of a programmable mark-up language

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

2 Useful references

2.1 Books

Lamport [1] 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.

Kopka and Daly [6] A comprehensive reference; it covers everything except the many add-on packages. Most people use this as their primary reference.

Mittelbach et al. [7] A guide to many of the most useful add-on packages and classes.

Goosens et al. [8] A slightly dated guide to packages for graphics.

Goosens et al. [9] A slightly dated guide to packages for adding hyperlinks, and producing PDF and HTML from L^AT_EX 2_ε.

2.2 Papers

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

The useful *general* papers are:

The Not So Short Introduction to L^AT_EX 2_ε [10]

Available in several languages.

¹CTAN, <http://www.ctan.org/>.

²<http://www.tug.org/texlive/>

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

2 Useful references

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

Math mode [11] A detailed explanation of typesetting mathematics in \LaTeX 2 ϵ .

The Comprehensive \LaTeX Symbol List [12] An enormous list of symbols and how to make them.⁴

Packages in the ‘graphics’ bundle [13] A bit out of date (it does not describe PDF extensions), but a useful introduction.

Hypertext marks in \LaTeX [14] 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 [15]

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.

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

TeXdoc Online: TeX and LaTeX Documentation <<http://texdoc.net/>> A fast way of finding documentation for T_EX-related things.

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

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

The L^AT_EX Project <<http://www.latex-project.org/>>

The centre of the L^AT_EX project.

The T_EX 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 T_EX, L^AT_EX and all classes and packages used in the example must be given.

Motivation <<http://www.slideshare.net/LianTzeLim/latex-more-than-just-academic-papers-and-theses>>

An excellent set of slides introducing what L^AT_EX can do for you, by Lian Tze Lim.

The beauty of L^AT_EX <<http://nitens.org/taraborelli/latex>> A page describing typographic advantages of T_EX-based systems over common competitors.

A wiki for L^AT_EX 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 L^AT_EX 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 T_EX (*not* L^AT_EX 2_ε) source code (especially mathematical source code) to see what the type-set version looks like.

2.4 Video

Marc van Dongen's Introduction to L^AT_EX <<http://youtu.be/s04nZHtl8jg>> Fifteen minutes and 50 seconds of advertisement and gentle introduction to the working model and basic features.

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).

$\text{\LaTeX} 2_{\epsilon}$ source may be created using any editor. Several editors have support for \TeX and $\text{\LaTeX} 2_{\epsilon}$, 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 $\text{\LaTeX} 2_{\epsilon}$ 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

3 *The L^AT_EX edit-process cycle*

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

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

A brief guide to using L^AT_EX 2_ε on (some of) the department's systems is given in Appendix C.

Part II

Concepts of L^AT_EX 2_ε

In this part of the document I briefly review some of the main concepts of L^AT_EX 2_ε documents.

This is *not* a comprehensive guide to L^AT_EX 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 $\text{\LaTeX} 2_{\epsilon}$ source file

The layout of a normal $\text{\LaTeX} 2_{\epsilon}$ document description is given in Listing 4.1.

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 Listing 4.1) 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 chapter 7).

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 $\text{\LaTeX} 2_{\epsilon}$ 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 Listing 5.1. 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 Listing 5.1.

Some classes, such as UoYCSproject, have a larger collection of declarations. (The declarations made available by UoYCSproject are given in Table 7.1.)

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 $\text{\LaTeX 2}_{\epsilon}$ 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 section 7.3.

Packages are loaded with the command `\usepackage{package name}`. An example is given in Listing 5.2 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 chapter 2 and Appendix A 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 $\text{\LaTeX}_{2\epsilon}$ 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 section 7.3.

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 Listing 5.3. 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 subsection 6.3.1).¹

¹If 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 T_EX parlance); `\newcommand` defines a command whose parameters *may* include paragraph breaks (‘long’ parameters in T_EX parlance). The ‘starred’ form is almost always the appropriate one.

As an example, Listing 5.4 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 Listing 5.5 and Listing 5.6. The second definition of `\msg` typesets the call ‘`\msg{A}{B}{M,K(A,B,N)}`’ as

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

while the third typesets it as

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

The `\newcommand` commands will report an error if the command name is already defined (possibly in the environment). You can *redefine*

```

\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

a command by using `\renewcommand` and `\renewcommand*`. There are other subtle variations on command definition, including the ability to define commands with one optional parameter. For these, see a standard book on \LaTeX 2 ϵ , such as those listed in section 2.1.²

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 Listing 5.7), for quotations (see Listing 5.8; 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 Appendix D), and for arranging formulæ (the `array` environment in Listing 5.6).

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.

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

As \cite{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 themselfe 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\—takammil\—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 himself promptly 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, \cite{P~627, Lines 8—16}{Joyce:FW}:

\begin{quotation}\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 thousandsthee. Lps. The keys to. Given! A way a lone a last a loved a long the

\end{quotation}

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 Listing 5.9 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 Listing 5.10.

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 Listing 5.11 (This protocol is due to Otway and Rees [16]; the notation $\{M\}_K$, marked up as `\enc{K}{M}`, means the encryption of M under symmetric key K). The typeset version is Protocol 1 on Page 48.

```

\newcounter{msgnumber}
\newenvironment*{protocol}
{ % begin code
  \setcounter{msgnumber}{0}%
  \newcommand*{\msg}[3]{%
    \refstepcounter{msgnumber}\thmsgnumber&##1&##2&##3}
  \newcommand*{\next}{\ }
  \begin{math}\displaystyle%
    \begin{array}{r@{.}\quad}l@{\rightarrow}l@{\;:\;}\l{}%
  \end{array}
  \end{math}%
}
{ % 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 Listing 6.1.

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 Listing 6.1. 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

```
% 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

the table of contents and the running headings. For example, see Listing 6.2.

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 Listing 6.1 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 re-processing that chapter. See the standard documentation.

<p>Here is some text incorrectly placed in math mode — note how different it is from paragraph mode:</p> <p><i>Here is some text incorrectly placed in math mode — note how different it is from paragraph mode.</i></p>
--

Figure 6.1: The result of treating text as mathematics

6.3 Text elements

6.3.1 Modes

$\text{\LaTeX} 2_{\epsilon}$ 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,

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 $\text{\LaTeX} 2_{\epsilon}$ 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 $\text{\LaTeX} 2_{\epsilon}$ will report an error. The reverse mistake—to place text in a math mode—results in ugly output; see Figure 6.1.

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 Listing 5.8, 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.

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

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

6.3.3 Characters

Reserved characters

There are some characters which are reserved and may not be used in text. These are listed in Table 6.1, together with how to make them if you really need them.

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 subsection 6.3.7.

Dashes

Another class of characters that sometimes causes confusion are the various dashes. See Table 6.2.

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 L^AT_EX 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

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

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 ‘ \TeX is 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 from thinking it is at a sentence end when it is not. \LaTeX (because \TeX does) treats a space as an

Series	<code>\mdseries</code>	Medium Series
	<code>\bfseries</code>	Boldface Series
Family	<code>\rmfamily</code>	Roman Family
	<code>\sffamily</code>	San Serif Family
	<code>\ttfamily</code>	Typewriter Family
	<code>\upshape</code>	Upright Shape
Shape	<code>\itshape</code>	<i>Italic Shape</i>
	<code>\slshape</code>	<i>Slanted Shape</i>
	<code>\scshape</code>	SMALL CAPS SHAPE

Table 6.3: Font attribute declarations

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.)

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 Table 6.3. 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 Table 6.4. (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.*

<code>\tiny</code> abcXYZ	<code>\scriptsize</code> abcXYZ	<code>\footnotesize</code> abcXYZ	<code>\small</code> abcXYZ
<code>\normalsize</code> abcXYZ	<code>\large</code> abcXYZ	<code>\Large</code> abcXYZ	<code>\LARGE</code> abcXYZ
<code>\huge</code> abcXYZ	<code>\Huge</code> abcXYZ		

Table 6.4: Font size declarations

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

L^AT_EX_{2 ϵ} 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). L^AT_EX_{2 ϵ} 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 Listing 6.3.

¹The 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

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 (section 2.1).

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.

²Actually, 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.

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 Listing 5.8.

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 Listing 5.8 and Listing 5.9 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 [17] 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 section B.6.

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	<code>\citep{Joyce:FW}</code>	generates [18]
	<code>\citep[\S4]{Joyce:FW}</code>	generates [18, §4]
	<code>\citep[see][\S4]{Joyce:FW}</code>	generates [see 18, §4]
Textual	<code>\citet{Joyce:FW}</code>	generates Joyce [18]
	<code>\citet[\S4]{Joyce:FW}</code>	generates Joyce [18, §4]
	<code>\citet[see][\S4]{Joyce:FW}</code>	generates Joyce [see 18, §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, Table 7.1), *figures* (for example, Figure 6.1) and *listings* (for example, Listing 5.8). 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 section 2.1).

Each float can have a symbolic label for use by L^AT_EX 2_ε’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 Table 6.1 and Table 7.1. There are also packages to give tabular environments with extra functionality. See a standard reference (section 2.1).

6.3.10 Theorem-like environments

L^AT_EX 2_ε allows you to define special series of named and numbered paragraphs, called theorem-like environments. The canonical examples

```

\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

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 Listing 6.4 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 be to roll the protocol and prot environments together; I have separated them here for illustration.) As an example here is Listing 5.11, 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

$\text{\LaTeX} 2_{\epsilon}$ has superb type-setting facilities for mathematics (see Figure 6.2). For complex work the $\mathcal{A}_{\mathcal{M}}\text{\TeX}$ packages (developed by the American Mathematical Society) will handle everything you could possibly need. Most people only need the basic, pre-loaded $\text{\LaTeX} 2_{\epsilon}$ facilities.³

³There is a school of thought that mistakes were made in the basic, pre-loaded $\text{\LaTeX} 2_{\epsilon}$ facilities, particularly the `eqnarray` environment. The solution proposed by this school

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

Figure 6.2: An example of mathematical type-setting. (Orthogonality in Chebyshev polynomials; T_n is the n th 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

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.

The subject is too complex to discuss here, and the standard references (section 2.1) should be consulted.

6.3.12 Cross references

$\text{\LaTeX} 2_\epsilon$ 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 Listing 6.5, 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 Listing 6.6.)

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:

- `\ref*`, which does *not* make the internal hyperlink.

is to always load and use the $\mathcal{A}\mathcal{M}\mathcal{S}\text{\LaTeX}$ packages.

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

- \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 $\text{\LaTeX} 2_{\epsilon}$ can either be

- drawn within $\text{\LaTeX} 2_{\epsilon}$ ’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 best⁴).

See the documentation given in chapter 2.

⁴There 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 2_{ϵ} 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 [19] (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 protrusion 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 Table 7.1.

¹You must print a document of class UoYCSProject 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
\ACS	—	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`, `\ACS`, `\SWE`, `\MIT`, or `\SCSE` (`\MIP` and `\GTC` are 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.

²Under Unix 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 section 7.2.

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 *Local Definitions*).

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 `\citert` for citations as text. See subsection 6.3.7.

The command `\cite` is defined to be the same as `\citert` (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 subsection 6.3.12.

Bibliography

- [1] L. Lamport, *LaTeX: A Document Preparation System*, 2nd ed. Reading, Mass.: Addison Wesley, 1994.
- [2] D. E. Knuth, *The TeX Book*. Addison Wesley, 1984.
- [3] —, *The Art of Computer Programming*, 3rd ed. Reading, Mass.: AddisonWesley, 1997, vol. 1: Fundamental Algorithms.
- [4] —, *The Art of Computer Programming*, 3rd ed. Reading, Mass.: Addison Wesley, 1998, vol. 2: Seminumerical Algorithms.
- [5] —, *The Art of Computer Programming*, 2nd ed. Reading, Mass.: Addison Wesley, 1998, vol. 3: Sorting and Searching.
- [6] H. Kopka and P. W. Daly, *A Guide to LaTeX: Document preparation for beginners and advanced users*, 3rd ed. Addison Wesley, 1999.
- [7] F. Mittelbach, M. Goossens, J. Braams, D. Carlisle, and C. Rowley, *The LaTeX Companion*, 2nd ed. Addison Wesley, Apr. 2004.
- [8] M. Goosens, S. P. Q. Rahtz, and F. Mittelbach, *The LaTeX Graphics Companion: Illustrating documents with TeX and PostScript*. Addison Wesley, 1997.
- [9] M. Goosens, S. P. Q. Rahtz, E. M. Gurari, R. Moore, and R. S. Sutor, *The LaTeX Web companion: Integrating TeX, HTML and XML*, ser. Addison-Wesley series on tools and techniques for computer typesetting. Addison Wesley Longman, 1999.
- [10] T. Oetiker, H. Partl, I. Hyna, and E. Schlegl, *The Not So Short Introduction to LaTeX 2_ε*, Dec. 2002, available from CTAN and the TeX Live distribution.
- [11] H. Voß, “Math mode,” Apr. 2007, available from CTAN and the TeX Live distribution.

Bibliography

- [12] S. Pakin, *The Comprehensive L^AT_EX Symbol List*, Sep. 2005, available from CTAN and the T_EX Live distribution.
- [13] D. P. Carlisle, *Packages in the ‘graphics’ bundle*, Jan. 1999, available from CTAN and the T_EX Live distribution.
- [14] S. Rahtz and H. Oberdiek, *Hypertext marks in L^AT_EX: a manual for HYPERREF*, Jul. 2003, available from CTAN and the T_EX Live distribution.
- [15] M. Kohm and J.-U. Morawski, *The KOMA-Script bundle*, Apr. 2003, available from CTAN and the T_EX Live distribution.
- [16] D. Otway and O. Rees, “Efficient and timely mutual authentication,” *Operating Systems Review*, vol. 21, no. 1, pp. 8–10, Jan. 1987.
- [17] M. Shell, *How to Use the IEEEtran BibT_EX Style*, IEEE, 2008, available from CTAN and the T_EX Live distribution.
- [18] J. Joyce, *Finnegans Wake*. London: Faber and Faber, May 1939.
- [19] W. Schmidt, *Using common PostScript fonts with L^AT_EX*, PSNFSS version 9.2 ed., Sep. 2004, available from CTAN and the T_EX Live distribution.

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 T_EX-Live release they can be obtained from the Common T_EX 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.

A Packages not pre-loaded that you may find useful

movie15 To insert moving images in the PDF. Particularly useful for presentations of physical artefacts (see section A.2).

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 $\text{\LaTeX} 2_{\epsilon}$ 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 $\text{\LaTeX} 2_{\epsilon}$ mode. Suitable for all flavours of $\text{\LaTeX} 2_{\epsilon}$. 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 $\text{\LaTeX} 2_{\epsilon}$ packages for preparing slides; a good list may be found at <http://www.miwie.org/presentations/>.

The main advantage of using a $\text{\LaTeX} 2_{\epsilon}$ -based solution for presentations is being able to re-use your $\text{\LaTeX} 2_{\epsilon}$ 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 L^AT_EX 2_ε ‘Gotchas’

There are just a few things that trip up a newcomer to T_EX, L^AT_EX 2_ε and BibT_EX.

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 ‘L^AT_EX 2_εis 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 ‘L^AT_EX 2_ε 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

T_EX 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

¹Some 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 section 6.3.3.

B.3 Wrong type of dash

There are four different types of dash available in T_EX-based systems. The different dashes have different uses: see section 6.3.3 for a discussion.

B.4 Wrong type of quote

In T_EX-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 subsection 6.3.6.

B.5 ‘Fragile’ commands in ‘moving’ arguments

B.5.1 Problem

Some arguments to L^AT_EX 2_ε 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 L^AT_EX 2_ε 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 Listing 6.5.)

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 L^AT_EX on the departmental systems

C.1 Under GNU/Linux

The default departmental T_EX-and-friends installation is the latest T_EXLive 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 T_EX 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 T_EXLive’.

To run L^AT_EX 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 L^AT_EX 2_ε — see chapter 3.

If you want to use AUC_TE_X 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 Listing 5.8 typeset

The typeset version of Listing 5.8 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 Joyce [18, 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 themselfe 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 (bababadalgharaghtakamminarronnkonnbronntonnerronntuonnthunntrovarrhounawnskawntooohooorderenthurnuk!) 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 promptly 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, [18, 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

D Listing 5.8 typeset

through grass behush the bush to. Whish! A gull. Gulls. Far calls.
Coming, far! End here. Us then. Finn, again! Take. Bussoftlhee,
mememormee! Till thousandsthee. Lps. The keys to. Given! A way
a lone a last a loved a long the
