Table of Contents

1 Error Messages

Some error messages will be `latex` messages, others `tex` messages.

When we run `latex`, if we get an error, we enter debugging mode.

Error 1:
undefined control serquence means latex didnt recognise the command at the `?` prompt we may type `x` to quit debugging, `h` for help.

2 Introduction

2.1 Compilation

source code -> `pdflatex` -> `.pdf`
source code -> `latex2html` -> `.html`

2.2 CTAN

The CTAN provides classes and packages made by the community, ranging in complexity and documentation. We can only consider a handfull.

2.3 Documentation

Use `texdoc` to search for documentation when exploring:

```
texdoc <package>
```

`texdoc -l <package>` lists all available documentation

2.4 Further Resources

- List of TeX Frequently Asked Questions
- Latex: A document preparation system - Leslie Lamport
- A guide to latex
- The latex companion
- The latex graphics companion
- The latex web companion
- Latex beginner's guide
- Latex and friends
- CTAN

3 Basics

3.1 Elementary Knowledge and Vocabulary

3.1.1 File Types

3.1.1.1 Class File

`.cls` defines a particular type of document, by setting its:

- page layout
- heading styles
- commands
- environments

3.1.1.2 Style File

Packages (`.sty` files) (re)define commands

Packages are imported via: `\usepackage[<<options>>]{<package>}`
Applicable class options will also be passed to the package

To learn about new packages, see Section 2.3 [Documentation], page 2.
To download and install packages, <Ref Appdx A>.

3.1.1.3 Main

source

The source code refers to all the text (including latex commands) that make up a document, saved with extension `.tex`.
This may be split across several files.

output

The `latex` application reads source code and creates a typeset document, the output file.
See Section 2.1 [Compilation], page 2, for commands that compile `latex` source code into the various output formats.

DVI

`tex` originally created `dvi` files, which can be converted to postscript.
`pdf`s are a similar file format, with extra facilities for embedded images and rotation. `pdf`s are now the default choice over `dvi`s.

`pdftex` allows us to create `pdf`s from `tex`.

3.1.1.4 Helper

Other types of file may be created in certain situations.

aux

Contains info for cross-referencing.
See detailed discussion of cross-referencing and multiple runs.

synctex

Enables jumping between corresponding points of source code and output file.

log

toc

lot

3.1.2 Development Environments

3.1.2.1 vim

3.1.2.2 Emacs

3.1.2.3 Front-End

A front-end is a piece of software that provides facilities for:

- writing source code
- passing source code to a latex application
- viewing the output file

3.1.3 CL Commands

- `texdoc`
- `pdflatex`
- `latex`
- `bibtex`
- `makeindex`
- `xindy`
- `kpsewhich`

3.1.4 Latex Commands

commands tell latex what to do at a particula rpoint in a document commands can take the following basic forms:

3.1.4.1 Modal-vs-Token

Modal commands are effective for the remainder of their block. They can be localised with:

- Standard grouping - `{\itshape <text>}`
- Environment grouping - `\begin{itshape}<text>\end{itshape}`

Token commands operate on a set number of mandatory arguments/tokens following the command. For example, token commands with a mandatory argument take the next token as their argument. Usually, this token is a `{}` block.

Note

Commands expecting no argument interpret `{}` as simply a block occuring after the command was made.
This can be used to create 'meaningful' whitespace after such a command. eg `\lambda{}`.

3.1.4.2 Control Words

Control words take the form `\[AZaz]*`.
A starred command is an asterisked control word.

3.1.4.3 Control Symbols

A control symbol is a `\` followed by a non-alphabet symbol.
Because we know the length of a control symbol is one, we don't need a space to signal the end. Therefore, whitespace after a control symbol is significant.

Starred control symbols are a special type of control symbol.
For example, `\\*` forces line break that cannot go over a page.

3.1.4.4 Character Sequences

Some sequences of characters form an instruction of their own.
For example:

- `ffi` is the `ffi` ligature
- `!'` is the upside down `!`

3.1.4.5 Internal Commands

Internal commands are like control words, with an `@` character in the name. They are private to class files and packages.

`@` takes on special meaning when the file is invoked by `\documentclass` or `\usepackage`. eg. `\c@section`, the internal representation of the section counter (in a class file or package), is the command `\c` that takes the character `@` as its argument, followed by `section`.

3.1.4.6 Short and Long Commands

A long command's argument may contain a paragraph break.
A short command's argument may not contain a paragraph break.
Short commands can automatically test for forgotten braces, so are preferable cet. par.

3.1.4.7 Declarations

A declaration is a command that affects the document from that point onwards.
Declarations do not themselves produce text, and can generally be localised by grouping.

3.1.4.8 Arguments

Some commands take mandatory or optional arguments.
This gives `latex` additional information in order to carry out the command.
The first token (non whitespace) following the command is the first argument. Hence we usually specify the argument as a group.
An empty (mandatory) argument is specified with {}.

mandatory

`\footnote` is a control word, taking one mandatory argument, that specifies the contents of the footnote.
`\chapter` is a control word that starts a new chapter, taking the chapter name as a mandatory argument.

optional

Optional commands are enclosed in `[]`.
Commands with both optional and mandatory arguments generally take the optional ones first.

For example, the control symbol `\\` takes an optional argument controlling the spacing between the lines: `\\[1pt]`.

moving fragile

A moving argument is an argument whose contents are copied to multiple parts of the document.
For example, the arguments to `\section` are moving arguments: they appear in both the table of context and at the beginning of the section.

Commands are considered 'fragile' if they can mess things up when used inside a 'moving argument'.
For example, `\footnote` is a fragile command because when used within a moving argument it will cause an error.

In particular, `\section{Section 1\footnote{test}}` causes problems.
We can manually fix the error using `\section{Section 1\protect\footnote{test}}`.

A robust command is a command that is not fragile.

3.1.4.9 Preamble

After the `\documentclass[options]{class}` command, and before `\begin{document}` (the start of the document environment), we have the preamble.

Most commands must come within the document environment, including all commands that generate text (eg `\maketitle`).
Some commands can be within the document or the preamble (eg `\title`).
A few commands can only come within the preamble (eg `\usepackage`).

3.1.5 Interpretation of Source

3.1.5.1 Grouping

A segment of code may be grouped within `{}`.
Most commands are local to their group.
Environments form an implicit scope.

3.1.5.2 Whitespace

Anything from `%` up to and including `EOL` is ignored.
Multiple whitespace is treated as a single whitespace.
A line consisting of only whitespace is treated as a `\par`.
Whitespace at the beginning of a line is ignored.

Hence:

```
Foo%
Bar
```

Produces:

```
FooBar
```

3.2 Document Structure

3.2.1 Abstract

Abstract abstracts are created in the abstrac environment some class files dont have an abstract environment the abstract environment should go after the \maketitle command

3.2.2 Appendix

Appendices start the appendix with \appendix then use sections normally to fill the appendix

3.2.3 Table of Contents

TOC \tableofcontents using all the sectioning commands in the document use it after \maketitle KOMA-Script classes have the optional argument toc=graduated(default)|flat as latex processes source code sequentially, it knows not the sectioning when it reads the toc as latex reads the sectioning, it writes a .toc file when latex reads \tableofcontents, it reads in the .toc file hence, two runs are required to fully update the toc

starred sections need to be manually added to the toc if desired the `\addcontentsline{<file format>}{<section unit>}{<text>}` is used for this purpose eg. `\chapter*{Acknowledgements}` `\addcontentsline{toc}{chapter}{Acknowledgements}` chapter arg ~ \chapter

3.2.4 Bibliography

Bibliography external tools such as bibtex and biber are used to automatically generate bibliography environments their use is covered in the next book for now, lets consider manually creating a bibliography environment

```
\begin{thebibliography}{<widest tag>}
\bibitem[<<tag>>]{<unique>}
\end{thebibliography}
```

depending on the document class, either \bibname prints the word Bibliography or \refname prints the word References thebibliography environment automatically invokes either \bibname or \refname to print the header, and then prints the bibitems

most classes dont add bib to TOC KOMA-Script option bibliography=totoc|totocnumbered

`\cite[<<text>>]{<unique>}` cites the reference

3.2.5 Titlepage

Titlepage requires at least the following info to be entered: `\author{}` `\title{}` They can also facilitate more info: `\date{}` if not provided, current date used by default, use empty `\date{}` to suppress KOMA-Script

```
\titlehead{}
\subject{}
\subtitle{}
\publishers{}
```

These commands do not print to the document, and hence may be placed in the preamble Useful commands in this context: `\and` for specifying multiple authors `\thanks{<text>}` for a special footnote `<text>` is a moving argument

3.2.6 Sectioning

Chapters, sections, etc.

```
\part[<<short title>>]{<<title>>}
\chapter[<<short title>>]{<<title>>}
\section[<<short title>>]{<<title>>}
\paragraph[<<short title>>]{<<title>>}
\subparagraph[<<short title>>]{<<title>>}
```

the arguments to the above are moving arguments the \parahraph commands are like unnumbered \subsubsubsections the <<short title>> is used in the TOC and page header

to facilitate consistent font styles for section headers, use \addtokomafont{<element>}{<commands>} eg \addtokomafont{section}{\rmfamily\bfseries}

KOMA-Script minisec \minisec{<heading>} provides an unnumbered independent heading

3.3 environments

An environment has the following form: \begin{environment} <block>\end{environment}

The interpretation of the block of code is dependent on the environment.

Some environments supply commands that may only be used within that environment.

3.4 Boxes and Minipages

tex views a page as a box. Boxes are built out of smaller boxes glued together.

* Boxes and line breaks

a box may not contain a line break \mbox{<text>} makes a box containing <text>, and <text> will not be allowed to span a line break

* Character boxes

The smallest box is for a single character:

```
 ____
|    |  height
|____|
|____|  depth
width
```

Note: ligatures get a single character box

* Horizontal boxes Horizontal boxes are boxes that can be placed inline with a line of text.

* Tabular environment

a tabular environment is a relatively complicated type of box it may be categorised as a horizontal box \tabular[<<pos>>]{<column specifiers>} <<pos>> determines the vertical alignment (t|c|b) of the inline box

* minipage environment the minipage environemtn is another type of inline box \begin{minipage}[<<pos>>][<<height>>]{<width>} eg \begin{minipage}[t]{0.4\linewidth} \parbox[<<pos>>][<<height>>]{<width>}{<text>} is a corresponding block-command, but as it is not an environment, it is more restricted than minipage (cannot do footnotes, list environments, etc)

* framed boxes

Framed boxes `\framebox[<<width>>][<<align-tcb>>]{<text>}` treats <text> as a box of width <width> and puts a frame around it `\fbox{<text>}` ~ `\framebox{<text>}` `\usepackage{fancybox}` `\shadowbox{<text>}` `\doublebox{<text>}` `\ovalbox{<text>}` `\Ovalbox{<text>}` advanced graphical frames, pgf, tikz

3.5 Cross Referencing

Cross-referencing `\label{<unique>}` should be placed at the point you want to label `\ref{<unique>}` is then used to refer to the chapter/section number containing the label `\pageref{<unique>}` is used to refer to the page containing the label

Note: dont start a line with a number - use unbreakable space - Section~\ref{sec:results}

You can use labels and refs within an enunmerate environment to refer to steps in the numbered list

similar to TOC, the refs may be invoked before the labels latex reads from the .aux file, needs two runs undefined refs show up as ?? in output if you get an 'undefined reference' warning on a re-run, then you have used a \ref to a \label that does not exist

rare situations involving cross referencing and TOCs can require three runs latexmk is a perl script that runs latex the required number of times

3.6 Multilingual Support

Multi-lingual support

`\usepackage[french, english]{babel}` last in list is the default `\selectlanguage{<language>}` `\begin{otherlanguage}{<language>}` `\foreignlanguage{spanish}{<text>}` use if only a short section of a different lagnuage needed `\iflanguage{english}{language is english}{language is not english}` check which language we are using

4 Intermediate

4.1 Details

4.1.1 accents

`\'‘^"utdr=.~vHcb{<object>}  m.a. of length 1`

inputenc and fontenc packages can be used to enable typing accents directly. but it is more hassle than it is worth for me

4.1.2 inter-secntence-spacing

french spacing uses 1 standard space
english spacing uses 1 en-space (slightly larger than 1 standard space)
tex defaults to english, which is better
\frenchspacing \nonfrenchspacing declare toggle french spacing

if we have a . ? or !, but it is not at the end of a sentence, then we must escape the en space
eg e.g.\ like this
if a we have a capital letter followed by . ? or !, latex assumes it is not the end of the sentence, so we must escape it
eg Yesterday, I saw my G.P\@.

Note: in a typewriter font, \nonfrenchspacing uses a double-space to simulate the en space

4.1.3 special-characters

Special characters

```
{}%$&_#^~\
\textbackslash
\textasciicircum
\textasciitilde
\pounds
\textregistered
\texttrademark
\copyright
\textbullet
\textquestiondown ?‘
\textexclamdown !‘
\textemdash ---
\textendash --
\textunderscore \_
\textperiodcentered
\ldots
```

```
\slash /  (Note / wont allow line break at / but \slash will)
\${}#%&
\i
\j
\S  section symbol
\P  paragraph symbol
\dag
\ddag
\textgreater  (in math mode < > and | are verbatim, in text mode, it varies)
\textbar
\textquoteright '
\textquotedblleft ``
texdoc symbols
```

4.1.4 italic-correction

Slanted font declarations may benefit from an italic correction \/ at the end of their block Especially noticeable of part of a word is stressed by being slanted

4.1.5 ligatures-and-kerning

```
man\oe uvre
man\oe{}uvre
man{\oe}uvre (not recommended - can affect kerning)

\ae, \oe, \aa, \l, \o, \ss, fi, ffi, fl, ffl
```

4.2 Page style and numbering

Page Styles and Numbering `\pagenumbering{arabic|roman|Roman|alph|Alph}` modal command the scrbook class provides `\frontmatter` `\mainmatter` and `\backmatter` declarations, which swich formatting for books

`\pagestyle{empty|plain|headings|myheadings}` `\thispagestyle{...}` `\markboth{<left-head>}{<right-head>}` if class has 'two-side' option - allows seperate style for odd and even pages `\markright{<right-head>}` if class has 'one-side' option

4.3 Line breaks

Hyphenation line breaks tex has an excellent hyphenation algo designed for english
use babel package for foreign language hyphenation algos
if tex needs help to find a good hyphenation, we can provide this as
1. \hyphenation{gal-axy}
2. gal\-axy

4.4 Fonts

Fonts

Appearence determined by: family, shape, series/weight Computer Modern font families (Knuth) used in text mode: (sans-)serif, typewriter

Changing font

Modal commands: \rmfamily \sffamily \ttfamily \mdseries \bfseries \upshape \itshape \slshape \scshape \em \normalfont Token commands: \textrm \textsf \texttt \textmd \textbf \textup \textit \textsl \textsc \emph \textnormal single mandatory argument \textrm test affects token t \textrm {test} affects token {test} Modal commands expect an no argument

Note: \bf \md \it \sl \sc \sf \tt \rm are deprecated in favour of less troublesome alternatives

\normalfont sets family, series and shape to default \em toggles between \upshape and \slshape

font size base font size for document is 11pt KOMA-Script default then the following declarations (with no corresponding block commands) can set the relative font size \tiny \scriptsize \footnotesize \small \normalsize \large \Large \LARGE \huge \HUGE corresponding environments \begin{small}<text>\end{small} == {\small <text>} font environments may be nested

Changing default document fonts documents have three fonts: rm, sf, tt the default computer modern fonts are used we can load packages to change them: mathptmx changes rm helvet changes sf courier changes tt the default font family is rm Helvetica is larger than Times so if we use \usepackage{mathptmx} we should use \usepackage[scaled=0.9]{helvet} For more fonts: The LATEX font catalogue. http://www.tug.dk/FontCatalogue/.