Design of a document class for thesis writing

Thesis uploaded for the good of mankind, in fulfilment of the requirements for the degree of

Master of TEXnology

in

Thesis formatting

by

LATEX enthusiast
Society of Document Beautifiers
Indian Institute of TEXnology

under the supervision of

TEX activist
Department Against Word Processors
Indian Institute of TEXnology

Contents

List of figures

List of tables

1

Introduction

This is the user manual for the `thesis` document class, a template designed to simplify the task of thesis writing. The `thesis` class is largely modelled on the standard `book` class, and produces an output document with a similar structure. However, the internal workings of the `thesis` class are very different from the `book` class, and were designed to make it fully customisable with minimal effort. This guide contains usage instructions along with illustrations wherever necessary. Besides, this document itself was created under the `thesis` class, allowing the reader to see what a document compiled under this class looks like.

The main features implemented in this class have been summarised in the following list. Each of these features is covered in depth in the upcoming chapters.

- *Basic class options*

  Class options can be passed to set the paper and font sizes, choose from among one- and two-sided printing styles, and indicate whether new chapters are allowed to begin on left hand side pages. Chapter 2 covers this in depth.

- *Book-style structure*

  The output document has a book-like structure. It comprises of an opening portion containing the title page, a front portion composed of an abstract, table of contents, list of figures, etc., a main portion which consists of the actual content of the thesis, and a back portion, which contains the bibliography. Document structure is the subject of Chapter 3.

- *Detailed title page*

  The title page contains all details usually found on a thesis cover. This includes the project title, the expected degree, names and affiliations of the author and supervisor, and also the university logo. Chapter 4 describes the procedure for constructing the title page.

- *Automatic tables of contents*

  The table of contents, list of figures, and list of tables are generated automatically by conveniently invoking simple commands, just like in standard LaTeX document classes. Chapter 5 covers the commands for doing this.

- *Figure and table environments*

  As discussed in Chapter 6, separate environments are provided to create floating elements such as figures and tables. These are identical to the structures implemented in standard document classes.

- *List structures*

  The `thesis` class provides three types of list structures – bulleted lists, enumerations, and description lists. A separate environment is defined for creating each type of listing. Detailed information on this can be found in Chapter 7.

- *Technical blocks*

  A simple command is provided to define new environments for technical statements such as proofs, definitions, theorems, etc. Under standard LaTeX classes, this is done with the help of the `amsthm` package. However, the `thesis` class provides this feature as an integral component, eliminating the need for `amsthm`. This feature is covered in Chapter 8.

- *Full programmability*

  The `thesis` class was designed by keeping customisation in mind. Hence, the class file `thesis.cls` is accompanied by another file `thesis.clo`, which contains a comprehensive list of parameters. These parameters can easily be reprogrammed in order to alter the appearance of the output document. This interface, covered in Chapter 9, offers the user full control over *all* formatting rules, with very little effort.

- *Hyperlink compatibility*

  Under standard LaTeX classes, automatic hyperlinks (using the `hyperref` package) attached to unnumbered entries in the table of contents point to incorrect locations in the PDF document. This behaviour has to be corrected manually by invoking the `\phantomsection` command every time an unnumbered element is created. However, the `thesis` class makes this correction automatically whenever the `hyperref` package is used, taking this bookkeeping task off the user's shoulders.

2

Class options

A few basic parameters may be passed as options to the `\documentclass` command, at the beginning of the source text. With these options, the user can control the paper and font sizes, select from among one- and two-sided printing styles, and indicate whether new chapters are allowed to open on left hand side pages. Options are passed along with the `\documentclass` command as follows.

```
\documentclass[<options>]{thesis}
```

Here, `<options>` is a comma separated list of options to be applied. This chapter contains a detailed description of all the options defined by the `thesis` class, as well as the concept of default options.

2.1 Available options

The options defined by the `thesis` class can be grouped into four major categories – paper size, font size, printing style, and chapter opening options. The following is a description of the options available under each category.

2.1.1 Paper size options

One out of three predefined paper sizes may be selected for the document – A4, letter, and legal. The corresponding options to choose these respective paper sizes are as follows.

```
a4paper,letterpaper,legalpaper
```

2.1.2 Font size options

The `thesis` class recognises the three usual font size categories – 10pt, 11pt, and 12pt. Each category defines a set of exact font sizes for the standard commands `\Huge`, `\huge`, ... `\scriptsize`, `\tiny`. The desired font size category may be selected by passing one of the following options.

```
10pt,11pt,12pt
```

2.1.3 Printing style options

Both one- and two-sided printing styles are possible under the `thesis` class. In a two-sided printing style, odd-numbered pages appear on the right hand side of the printed book, and even-numbered pages on the left. In a one-sided document, all pages appear on the right hand side. The running page headers and footers are defined differently for left, and right hand side pages. As visible in this very document, odd page numbers are placed on the right side of the page, but even page numbers are placed towards the left. In one-sided printing, all pages are right hand side pages, and will have identical header and footer formats. The printing style may be selected by applying one of the following two options.

```
oneside,twoside
```

2.1.4 Chapter opening options

New chapters may be allowed to either open only on right hand side pages, or on any pages. Both these schemes are equivalent in one-sided documents. In a two-sided document however, requiring a new chapter to open on the right hand side means that it can only start on an odd-numbered page, skipping an additional blank page if necessary. The user may specify their preference by applying any one of the following two options.

```
openright,openany
```

2.2 Default options

Options are applied in two successive steps. First of all, a set of default options is applied, regardless of the options passed by the user. After this, the user-specified options (if any) are applied, overriding any of the default options if necessary. The following is the set of default options, which may therefore, be eliminated from the source text.

```
a4paper,10pt,oneside,openright
```

Thus, a two-sided document with all other parameters set to default, may be created by just passing the `twoside` option alone.

2.3 Option summary

The following is a summary of all options defined under the `thesis` class. Default options have been marked with an asterisk.

`a4paper`*	A4-sized paper.
`letterpaper`	Letter-sized paper.
`legalpaper`	Legal-sized paper.
`10pt`*	10pt fonts.
`11pt`	11pt fonts.
`12pt`	12pt fonts.
`twoside`	Two-sided printing style.
`oneside`*	One-sided printing style.
`openright`*	New chapters must begin on right hand side.
`openany`	New chapters may begin on any page.

3

Document structure

The `thesis` class implements a hierarchical document structure consisting of *chapters*, *sections*, *subsections*, and *sub-subsections*. Additionally, the document is also divided into four major portions – the *opening*, *front*, *main*, and *back* portions – that serve different purposes. The structure of chapters, and their subsequent sectioning is defined differently for these different portions.

3.1 Document portions

The document must begin with the *opening* portion. This should be followed by the *front*, *main*, and *back* portions in that order. The opening portion is meant to contain the title page. The front portion is composed of all the material that appears before the main content of the thesis begins. This includes the acknowledgements, abstract, table of contents, list of figures, list of tables, etc. The main portion contains the actual content of the thesis, divided into multiple chapters and appendices. The back portion is meant for the bibliography.

The following are the commands for creating the four document portions, along with detailed descriptions of these portions.

- `\opening`

  This command marks the beginning of the opening portion. Page headers and footers are not printing in this portion, and page numbering is not performed. The title page (see Chapter 4) can only be created in this portion. The opening portion is also a good place to put additional pages such as a dedication or a quote, as nothing else shall appear on these pages (like page headers and footers).

- `\front`

  This command marks the beginning of the front portion. It can be executed only if the `\opening` command has already been executed. Page headers and footers are printed in this portion, and page numbering is performed. Page numbers are printed as lower case Roman numerals. Chapters created in this portion are unnumbered and represent the abstract, table of contents, etc.

- `\main`

  This command marks the beginning of the main portion. It can be executed only if the `\front` command has already been executed. Page headers and footers are printed in this portion. Page numbering is restarted from 1, and page numbers are printed in the ordinary Arabic format. Chapters created in this portion are numbered and represent ordinary chapters as well as appendices that constitute the main content of the thesis.

- `\back`

  This command marks the beginning of the back portion. It can be executed only if the `\main` command has already been executed. Page headers and footers are printed in this portion. Page numbering continues from the main portion into this portion. Chapters created in this portion are unnumbered. The back portion will usually consist of only one chapter, and that is used to represent the bibliography.

Thus, the overall structure of the source text should be as follows.

```
\begin{document}
\opening
  <contents of opening portion>
\front
  <contents of front portion>
\main
  <contents of main portion>
\back
  <contents of back portion>
\end{document}
```

3.2 CHAPTERS

Chapters are the primary structures for document sectioning in the `thesis` class. A chapter can be created with the `\chapter` command, as follows.

```
\chapter{<chapter-name>}
```

Here, `<chapter-name>` is the name of the chapter being created. Depending on the document portion in which the above command is executed, different outcomes can be observe, as elaborated below.

- *Opening portion*

  A compilation failure occurs. Chapter creation is not allowed in the opening portion.

- *Front portion*

  An unnumbered chapter is created. The current page is skipped if not blank, and an unnumbered chapter heading containing the chapter name is created on the new page. If the `twoside` and `openright` class options were applied, then an additional page may be skipped to ensure that the new chapter begins on an odd-numbered page. An entry for the new chapter is added to the table of contents (see Chapter 5). Page headers are updated to indicate the current chapter. The front portion will usually comprise chapters like `\chapter{Abstract}`, `\chapter{Contents}`, `\chapter{List of figures}`, `\chapter{List of tables}`, etc.

- *Main portion*

  A numbered chapter is created. The current page is skipped if not blank, and a chapter heading containing the chapter name and number is created on the new page. If the `twoside` and `openright` class options were applied, then an additional page may be skipped to ensure that the new chapter begins on an odd-numbered page. An entry for the new chapter is added to the table of contents. Page headers are updated to indicate the current chapter.

In the main portion, one of two modes – *chapter* and *appendix* – may be active. Chapter numbers are printed in Arabic format in chapter mode, and alphabetically in appendix mode. The mode can be switched by using the `\appendix` command, described further below.

- *Back portion*

  An unnumbered chapter is created. The process is identical to that of chapter creation in the front portion. The back portion will usually consist of a single chapter, namely `\chapter{Bibliography}`.

As mentioned above, two modes – *chapter* and *appendix* – are possible in the main portion. Upon starting the main portion with the `\main` command, the chapter mode is automatically activated. The command `\appendix` may be invoked at any point in the main portion, to switch to appendix mode. This command can be run only once. Also, it is not possible to switch back from appendix to chapter mode (should this ever become necessary, you should realise that you have designed your document badly).

To sum up, the structure of the main portion in the source text should be similar to the following snippet.

```
\main
\chapter{<first chapter name>}
  <first chapter content>
\chapter{<second chapter name>}
  <second chapter content>
\appendix
\chapter{<first appendix name>}
  <first appendix content>
\chapter{<second appendix name>}
  <second appendix content>
```

3.3 Further sectioning

The `thesis` class offers three hierarchical levels of sectioning within a chapter. These are achieved by invoking the `\section`, `\subsection`, and `\subsubsection` commands. The syntax for each of these commands is similar to that of the `\chapter` command.

```
\section{<section name>}
\subsection{<subsection name>}
\subsubsection{<sub-subsection name>}
```

The above commands create a section, subsection, or sub-subsection header, and add a corresponding entry to the table of contents. The section, subsection, or sub-subsection thus created is numbered if created within a numbered chapter, and unnumbered if created within an unnumbered chapter.

It is only possible to proceed in steps of one hierarchical level at a time. This means that a section can only be created within a chapter, a subsection within a section, and so on. Not doing so will result in an "insufficient sectioning depth" error upon compilation (in standard LaTeX document classes, compilation succeeds but leads to absurd numbering).

3.4 Note on undefined commands

The `thesis` class does not define the familiar starred commands `\chapter*`, `\section*`, etc. Also, the commands `\part`, `\paragraph`, and `\subparagraph` are not defined, at least as of now.

4

Title page

The `thesis` class features a detailed title page which contains the project title, expected degree, names and affiliations of the author and supervisor, and the university logo. Under this document class, the title page can only be created in the opening portion (see Section 3.1).

4.1 Title page parameters

The procedure to create a title page is identical to that of standard LaTeX document classes. Values must be defined for a set of parameters, which are then put together by the `\maketitle` command to form the title page.

The following is a list of commands that set values for the different parameters required to generate the title page. It is necessary to run each of the following commands once before invoking `\maketitle`. The following commands are usually placed in the preamble of the source text, but this is not a strict rule to follow.

- `\title{<project-title>}`
  Sets the thesis project title to `<project-title>`. The project title appears at the top of the title page.
- `\purpose{<submission-purpose>}`
  Sets `<submission-purpose>` as the phrase indicating the purpose of submission of the thesis. This will usually be a phrase like "Thesis submitted in partial fulfilment of the requirements for the degree of". Following this phrase, the expected degree title (see next command) appears right below. The degree title should not be included in the phrase `<submission-purpose>`.
- `\degree{<expected-degree>}`
  Sets the expected degree title to `<expected-degree>`. The degree title must not include the academic field in which it shall be awarded. For instance, "Bachelor of Science" or "Doctor of Philosophy" are appropriate values for `<expected-degree>`, but "Bachelor of Science in Physics" is not. The academic field appears on a separate line and is specified by the next command in this list.

- `\subject{<academic-field>}`
  Sets the academic field in which the degree shall be awarded, to `<academic-field>`. For example, "Physics", "Aerospace engineering", etc. are appropriate values. Together with the degree title, this results in a phrase like "Bachelor of Science in Physics" split over multiple lines.
- `\author{<author-name>}`
  Sets the name of the author to `<author-name>`.
- `\authoraffil{<author-affiliation>}`
  Sets the author's affiliation to `<author-affiliation>`. For instance, the name of the author's department in the university could be specified as `<author-affiliation>`.
- `\mentor{<mentor-name>}`
  Sets the name of the project supervisor to `<mentor-name>`.
- `\mentoraffil{<mentor-affiliation>}`
  Sets the project supervisor's affiliation to `<mentor-affiliation>`. For instance, the name of the project supervisor's department in the university could be specified as `<mentor-affiliation>`.
- `\logo{<university-logo>}`
  Sets the command to generate the university logo, to `<university-logo>`. Thus, `<university-logo>` will itself be a command, usually containing an invocation of `\includegraphics`.

4.2 Generating the title

Once each of the commands listed in Section 4.1 has been run, the title page can be created by simply executing the `\maketitle` command. However, it must be kept in mind that `\maketitle` can only be called in the opening portion (see Section 3.1) of the document. Thus, the part of the source text that generates the title page must resemble the following snippet.

```
\title{<project-title>}
\purpose{<submission-purpose>}
\degree{<expected-degree>}
\subject{<academic-field>}
\author{<author-name>}
\authoraffil{<author-affiliation>}
\mentor{<mentor-name>}
\mentoraffil{<mentor-affiliation>}
\logo{<university-logo>}

\begin{document}
\opening
  \maketitle
\front
  \<front-matter>
\main
  \<main-matter>
\back
  \<back-matter>
\end{document}
```

5

Document contents

The `thesis` class provides commands to generate three basic listings of contents – the table of contents, list of figures, and the list of tables. The usual place to create these lists is the front portion (see Section 3.1) of the document. The corresponding commands are as follows.

- `\generatetoc`
  Lists out the *table of contents*. This includes all document sectioning structures described in Chapter 3 – chapters, sections, subsections, and sub-subsections. Numbered as well as unnumbered elements are listed out.
- `\generatelof`
  Lists out the *list of figures*. This includes all figures created using the `figure` environment. The `figure` environment is discussed in detail in Chapter 6.
- `\generatelot`
  Lists out the *list of tables*. This includes all tables created using the `table` environment. The `table` environment is discussed in detail in Chapter 6.

While these commands list out their corresponding document contents, they do not first navigate to a new page or create a heading. In order to create the necessary headings, and start each of these lists on new pages, the above commands must be invoked within separate chapters. Thus, an appropriate method to list out the document contents would be as follows.

```
\front
<any other content, such as abstract, acknowledgements, etc.>
\chapter{Contents}
  \generatetoc
\chapter{List of figures}
  \generatelof
\chapter{List of tables}
  \generatelot
```

6

Figures and tables

Figures and tables are created as *floating* structures (elements that treated separately from the main text in the document). We first discuss the basics of floating structures in Section 6.1. Following this, Sections 6.2 and 6.3 cover the processes of creating figures and tables respectively.

6.1 Floating elements

Floating elements are structures that will never be split over multiple pages. Also, they are not part of the regular stream of text, and are placed in a separately allocated part of the page. The compiler may choose from among various different positions to place a floating element – top of page, bottom of page, approximately where it appears in the source text, or on a separate special page for floats only. Each of these locations is represented by a corresponding *placement specifier* (see Table 6.1). The user may optionally pass one or more placement specifier while creating a floating element. This instructs the LaTeX compiler to choose from only those locations, while placing the floating element in question.

Each floating structure is implemented as a separate numbered environment (thus, the `thesis` class recognises two environments – `figure` and `table`). Every floating element has a *caption* that indicates this number, and also contains a short description written by the user. Also, floating elements can be created only within numbered chapters, as their numbering includes the chapter number.

Specifier	Placement position
`h`	Place float approximately *here* (same point it occurs in source text).
`t`	Place float at the *top* of the page.
`b`	Place float at the *bottom* of the page.
`p`	Create a special *page* just for floats and place float there.

Table 6.1: List of placement specifiers that can optionally be passed while creating a floating structure. The default set of specifiers passed is `tbp`.

The code to generate a floating element of any kind is as follows.

```
\begin{<envname>}[<placement>]
  <content>
  \caption[<captshort>]{<captlong>}
  \label{<labname>}
\end{<envname>}
```

Here, `<envname>` is the name of the floating environment (in the `thesis` class, this will be either `figure` or `table`). The `<placement>` argument represents the optional placement specifiers. If left unspecified, the default placement specifiers `tbp` are passed. The code to generate the actual content of the floating element (the image or table) must replace `<content>`. Two kinds of captions can be specified. A long caption `<captlong>` is what appears along with the floating element as the actual caption, and a short caption `<captshort>` goes into the list of figures or list of tables. A label `<labname>` can be attached for use with the `\ref` command.

6.2 FIGURES

Figures are created using the floating environment `figure`. The image can be created using the `\includegraphics` command from the `graphicx` package. Detailed information about the `graphicx` package can be found in its official documentation.

For example, the following code was executed to generate Figures 6.1 and 6.2.

```
\begin{figure}[p]
  \begin{center}
    \includegraphics[scale=0.075]{examplefigure1.jpg}
  \end{center}
  \caption[Short title for example figure 1]{
    This is a detailed description of example figure 1.
    It appears along with the figure, as a paragraph in the caption.
    The short title for this figure appears in the list of figures.
  }
  \label{fig:examplefigure1}
\end{figure}

\begin{figure}[p]
  \begin{center}
    \includegraphics[scale=0.3]{examplefigure2.jpg}
  \end{center}
  \caption[Short title for example figure 2]{
    This is a detailed description of example figure 2.
    It appears along with the figure, as a paragraph in the caption.
    The short title for this figure appears in the list of figures.
  }
  \label{fig:examplefigure2}
\end{figure}
```

6.3 Tables

Tables are created using the floating environment `table`. The tabular arrangement can be created using the `tabular` environment. Detailed information about the `tabular` environment can be found on wikibooks.

For example, the following code was executed to generate Tables 6.2 and 6.3.

```
\begin{table}[p]
  \begin{center}
    \begin{tabular}{|c|c|c|}
    \hline
    $P_{11}$ & $P_{12}$ & $P_{13}$\\
    \hline
    $P_{21}$ & $P_{22}$ & $P_{23}$\\
    \hline
    \end{tabular}
  \end{center}
  \caption[Short title for example table 1]{
    This is a detailed description of example table 1.
    It appears along with the table, as a paragraph in the caption.
    The short title for this table appears in the list of tables.
  }
  \label{tab:exampletable1}
\end{table}

\begin{table}[p]
  \begin{center}
    \begin{tabular}{|c|c|c|}
    \hline
    $Q_{11}$ & $Q_{12}$ & $Q_{13}$\\
    \hline
    $Q_{21}$ & $Q_{22}$ & $Q_{23}$\\
    \hline
    \end{tabular}
  \end{center}
  \caption[Short title for example table 2]{
    This is a detailed description of example table 2.
    It appears along with the table, as a paragraph in the caption.
    The short title for this table appears in the list of tables.
  }
  \label{tab:exampletable2}
\end{table}
```

FIGURE 6.1: This is a detailed description of example figure 1. It appears along with the figure, as a paragraph in the caption. The short title for this figure appears in the list of figures.

FIGURE 6.2: This is a detailed description of example figure 2. It appears along with the figure, as a paragraph in the caption. The short title for this figure appears in the list of figures.

P_{11}	P_{12}	P_{13}
P_{21}	P_{22}	P_{23}

TABLE 6.2: This is a detailed description of example table 1. It appears along with the table, as a paragraph in the caption. The short title for this table appears in the list of tables.

Q_{11}	Q_{12}	Q_{13}
Q_{21}	Q_{22}	Q_{23}

TABLE 6.3: This is a detailed description of example table 2. It appears along with the table, as a paragraph in the caption. The short title for this table appears in the list of tables.

7

Lists and enumerations

Three different types of lists can be created under the `thesis` document class – bulleted, numbered, and description lists. Each such list structure is implemented as a different environment. The environments that correspond to bulleted, numbered, and description lists respectively are `listing`, `enumeration`, and `description`.

7.1 Basic syntax

A list consists of multiple entries packed into one of the three list structures mentioned above. List entries are created by executing the `\item` command, followed by the corresponding text. In the output document, each list entry starts with a left-aligned label (which can be a bullet, a number, or a description title). A fixed margin is reserved on the left side for the label. The textual content appears after the label, and is formatted into paragraphs. The following syntax applies for creating any of the three kinds of lists. This snippet shows two entries, but there can be any number of them in general.

```
\begin{<envname>}[<labwidth>]
\item[<label1>]  <Text1>
\item[<label2>]  <Text2>
\end{<envname>}
```

Here, `<envname>` must be either `listing`, `enumeration`, or `description`. The optional label width `<labwidth>` is the margin reserved for entry labels. If unspecified, a default label width is applied. The individual labels for the two entries in the above snippet are `<label1>` and `<label2>`, and are also optional. For bulleted lists and enumerations, it is generally a good idea to leave them unspecified, and let the default labels appear. In description lists however, these are meant to be specified by the user as they represent description titles. The texts corresponding to the two entries in the above example are `<Text1>` and `<Text2>`, and may each contain multiple paragraphs.

List entries may themselves contain lists too, resulting in a nested list structure. Bulleted and numbered lists (created with the `listing` and `enumeration` environments) can be nested only up to 4 levels. Description lists on the other hand, can be nested indefinitely.

7.2 List examples

The following examples illustrate the creation of all three kinds of lists. Notice the change in label format at different levels in the bulleted, and numbered lists.

7.2.1 Bulleted list

```
\begin{listing}
\item   Item 1
        \begin{listing}
        \item   Item 1.1
        \item   Item 1.2
        \end{listing}
\item   Item 2
        \begin{listing}
        \item   Item 2.1
        \item   Item 2.2
        \end{listing}
\end{listing}
```

- Item 1
  - Item 1.1
  - Item 1.2
- Item 2
  - Item 2.1
  - Item 2.2

7.2.2 Numbered list

```
\begin{enumeration}
\item   Item 1
        \begin{enumeration}
        \item   Item 1.1
        \item   Item 1.2
        \end{enumeration}
\item   Item 2
        \begin{enumeration}
        \item   Item 2.1
        \item   Item 2.2
        \end{enumeration}
\end{enumeration}
```

1. Item 1
   a. Item 1.1
   b. Item 1.2
2. Item 2
   a. Item 2.1
   b. Item 2.2

7.2.3 Description list

```
\begin{description}[1.5cm]
\item[Item 1]   Description of item 1
\item[Item 2]   Description of item 2
\item[Item 3]   Description of item 3
\end{description}
```

Item 1 Description of item 1

Item 2 Description of item 2

Item 3 Description of item 3

8

Technical blocks

Technical blocks are a special feature implemented in the `thesis` class, for formatting technical statements like theorems, definitions, proofs, etc. Under the standard LaTeX document classes, the `amsthm` package provides this functionality. However, this package is not required while working with the `thesis` class.

8.1 Block environments

We use the term *block* to represent any chunk of text that constitutes a theorem, proof, definition, or any other type of technical statement. Each such type of technical statement must be associated with a separate *block environment*. For example, the user may define three environments – `theorem`, `proof`, and `definition` – to create a theorem block, proof block, or definition block. In order to easily define such block environments, the `thesis` class provides a special command `\newblock`, with the following syntax.

`\newblock{<envname>}[<flags>]`

Here, `<envname>` is the name of the new environment being defined. It is also the heading text for the newly defined block. Whenever a new block is created with the command `\begin{<envname>}`, the word `<envname>` (with its first alphabet capitalised) appears as a heading. If the block is numbered (which is controlled by the `<flags>` argument), the heading also contains the block number. The argument `<flags>` is a comma separated list of three options, one from each of the following three sets. The order in which these options are listed does not matter.

- `num, nonum`
  The `num` option makes `<envname>` a numbered environment, and `nonum` makes it unnumbered.
- `emph, noemph`
  The `emph` option italicises the text within the block, and `noemph` leaves it upright.
- `term, noterm`
  The `term` option prints a Q.E.D. symbol at the end of the block, and `noterm` avoids it.

8.2 Examples of blocks

In the following example, three block environments are defined to create theorems, definitions, and proofs. The \newblock command is invoked in the preamble of the source text.

```
\newblock{definition}[num,emph,noterm]
\newblock{theorem}[num,emph,noterm]
\newblock{proof}[nonum,noemph,term]
```

Having defined the three environments, they can be used inside the document, to create theorem, proof, and definition blocks. The following snippet creates a definition block, and two theorem blocks accompanied with corresponding proof blocks, as shown below it.

```
\begin{definition}
This is a simple definition.
It appears with a numbered heading of its own.
The definition text is italicised.
\end{definition}

\begin{theorem}\label{thm:theorem1}
This is a simple theorem.
Its format is similar to that of a definition.
\end{theorem}

\begin{proof}
This is the proof of Theorem \ref{thm:theorem1}.
It appears with an unnumbered heading.
The proof text is not italicised.
At the end of this proof, you shall find a Q.E.D.\ symbol.
\end{proof}

\begin{theorem}\label{thm:theorem2}
This is another simple theorem.
Its format is similar to that of a definition.
\end{theorem}

\begin{proof}
This is the proof of Theorem \ref{thm:theorem2}.
It appears with an unnumbered heading.
The proof text is not italicised.
At the end of this proof, you shall find a Q.E.D.\ symbol.
\end{proof}
```

Definition 8.1 *This is a simple definition. It appears with a numbered heading of its own. The definition text is italicised.*

Theorem 8.1 *This is a simple theorem. Its format is similar to that of a definition.*

Proof: This is the proof of Theorem 8.1. It appears with an unnumbered heading. The proof text is not italicised. At the end of this proof, you shall find a Q.E.D. symbol. ▪

Theorem 8.2 *This is another simple theorem. Its format is similar to that of a definition.*

Proof: This is the proof of Theorem 8.2. It appears with an unnumbered heading. The proof text is not italicised. At the end of this proof, you shall find a Q.E.D. symbol. ▪

9

Modifying class parameters

The `thesis` class was designed with the objective of creating a fully programmable template. The source code for this class consists of two files – `thesis.cls` and `thesis.clo`. The file `thesis.cls` contains all the fundamental macros, and is essentially the defining file for the `thesis` class. The `thesis.clo` file is a comprehensive list of parameters and simple macros, that can easily be modified to alter the appearance of the output. While the `thesis.cls` file is not supposed to edited at all, the file `thesis.clo` is just meant for editing. By modifying the definitions in `thesis.clo`, the user gets full control over all formatting rules defined by the `thesis` class.

The parameters defined in `thesis.clo` are grouped into various categories. In order to identify them easily, a simple naming convention has been adopted. Each macro name is of the form `\@<category-name>@<parameter-name>`. The following sections provide a category-wise description of all the parameters in `thesis.clo`. Each section starts by specifying the initial portion of the macro names (category part) and line numbers in the default version of `thesis.clo` where the corresponding parameters can be found.

9.1 Font sizes

Macro name prefix: `\@font@`
Lines: 12 – 57

- `\@font@xpt`
  Command run to define 10pt font sizes and corresponding paragraph spacing parameters.
- `\@font@xipt`
  Command run to define 11pt font sizes and corresponding paragraph spacing parameters.
- `\@font@xiipt`
  Command run to define 12pt font sizes and corresponding paragraph spacing parameters.

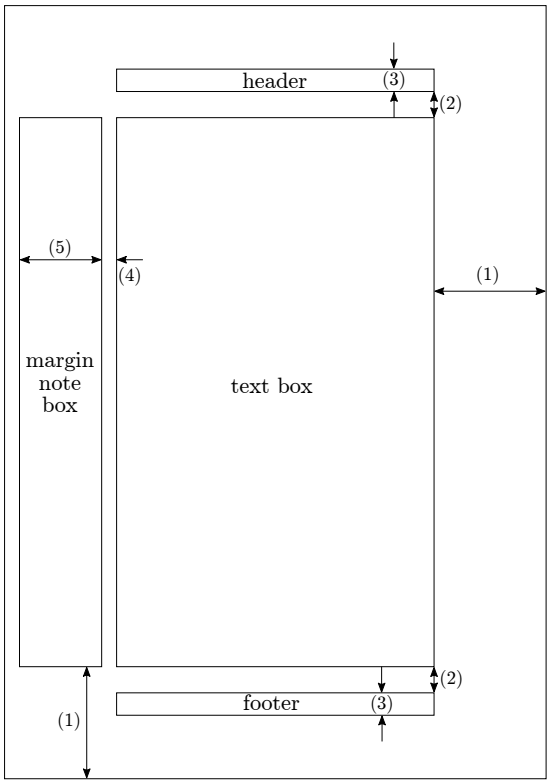

Figure 9.1: A page consists of four major fields – the text, margin note, header, and footer – as shown here. The main document text goes into the text box. Margin notes go into the margin note box, which can be on either the left, or right side of the page, depending on whether the page itself is a left hand side, or right hand side page. Page headers and footers are also allocated dedicated spaces on the page. The dimensions (1) to (5) that control the page layout are defined in Section 9.2.

9.2 Page dimensions

Macro name prefix: `\@pgdim@`
Lines: 58 – 66

- `\@pgdim@textmargin`
  Space between page borders and text box borders. Dimension (1) in Figure 9.1.
- `\@pgdim@headerseparation`
  Separation between text box and header/footer box borders. Dimension (2) in Figure 9.1.
- `\@pgdim@headerheight`
  Height of header, and footer boxes. Dimension (3) in Figure 9.1.
- `\@pgdim@marginnoteseparation`
  Separation between text box and margin note box borders. Dimension (4) in Figure 9.1.
- `\@pgdim@marginnotewidth`
  Width of margin note box. Dimension (5) in Figure 9.1.

9.3 PAGE DECORATIONS

Macro name prefix: \@pgdec@
Lines: 67 – 74

- \@pgdec@leftheader
  Command to generate the left hand side page header.
- \@pgdec@rightheader
  Command to generate the right hand side page header.
- \@pgdec@leftfooter
  Command to generate the left hand side page footer.
- \@pgdec@rightfooter
  Command to generate the right hand side page footer.

The control sequences \leftmark and \rightmark are often used within the definitions of the above macros. These are page markings whose values are set by the \markboth and \markright commands. The command \markboth{<lmark>}{<rmark>} sets the value of \leftmark to <lmark> and that of \rightmark to <rmark>. In the default version of thesis.clo, these commands are invoked by the \chapter command (see Section 9.4) so that \leftmark and \rightmark indicate the current chapter heading. Also, the values of \leftmark and \rightmark are used within the page headers. This is the mechanism by which the \chapter command updates the page headers.

9.4 DOCUMENT STRUCTURE

Macro name prefix: \@docstruc@
Lines: 75 – 169

In the following listing, every instance of <struc> must be replaced by chapter, section, subsection, or subsubsection.

- \@docstruc@<struc>numberlesstocskip
  Vertical space added to the table of contents whenever \<struc> is executed to create an unnumbered <struc>.
- \@docstruc@<struc>numberedtocskip
  Vertical space added to the table of contents whenever \<struc> is executed to create a numbered <struc>.
- \@docstruc@<struc>numberlesslofskip
  Vertical space added to the list of figures whenever \<struc> is executed to create an unnumbered <struc>.
- \@docstruc@<struc>numberedlofskip
  Vertical space added to the list of figures whenever \<struc> is executed to create a numbered <struc>.
- \@docstruc@<struc>numberlesslotskip
  Vertical space added to the list of tables whenever \<struc> is executed to create an unnumbered <struc>.
- \@docstruc@<struc>numberedlotskip
  Vertical space added to the list of tables whenever \<struc> is executed to create a numbered <struc>.

- \@docstruc@make<struc>numberless#1
  Command to generate an unnumbered <struc> heading with <struc> name #1.
- \@docstruc@make<struc>numbered#1
  Command to generate a numbered <struc> heading with <struc> name #1.

9.5 Title page

Macro name prefix: \@titlpg@
Lines: 170 – 205

- \@titlpg@create#1#2#3#4#5#6#7#8#9
  Command to create the title page with the 9 parameters #1 ... #9. The arguments correspond to the values defined by the commands in Section 4.1. They are described below in the same order in which they are passed to \@titlpg@create.
  1. Project title, as defined by the \title command.
  2. Purpose of submission, as defined by the \purpose command.
  3. Degree title, as defined by the \degree command.
  4. Academic field, as defined by the \subject command.
  5. Author name, as defined by the \author command.
  6. Author affiliation, as defined by the \authoraffil command.
  7. Project supervisor name, as defined by the \mentor command.
  8. Project supervisor affiliation, as defined by the \mentoraffil command.
  9. University logo creation command, as defined by the \logo command.

9.6 Document contents

Macro name prefix: \@doccnts
Lines: 206 – 257

In the following listing, every instance of <struc> must be replaced by chapter, section, subsection, subsubsection, figure, or table. Additionally, every instance of <cnts> must be replaced by "table of contents", "list of figures" or "list of tables" depending on what <struc> represents.

- \@doccnts@<struc>filler
  Symbol to be repeatedly printed to fill up the space between a <struc> entry and the corresponding page number in the <cnts>.
- \@doccnts@<struc>indent
  Amount of indentation added to a <struc> entry in the <cnts>.
- \@doccnts@<struc>numberwidth
  Horizontal space within a <struc> entry, reserved for the <struc> number. This space is created even for unnumbered entries, but is left blank.
- \@doccnts@<struc>pagenumberwidth
  Horizontal space within a <struc> entry, reserved for the page number.
- \@doccnts@<struc>fillerwidth
  Space between successive filler symbols in a <struc> entry.

- \@doccnts@<struc>entryfont
  Font used for the main text in a <struc> entry.
- \@doccnts@<struc>fillerfont
  Font used for the filler in a <struc> entry.
- \@doccnts@<struc>pagenumberfont
  Font used for the page number in a <struc> entry.

9.7 FIGURES AND TABLES

Macro name prefix: \@figtab@
Lines: 258 – 271

- \@figtab@captionaboveskip
  Vertical blank space above caption.
- \@figtab@captionbelowskip
  Vertical blank space below caption.
- \@figtab@captionmargin
  Horizontal blank space to the left and right of caption, added to the usual text margin.
- \@figtab@captionheadformat
  Font used in the caption heading.
- \@figtab@captionbodyformat
  Font used in the main caption body.
- \@figtab@figurelabeltext
  Caption heading text for figures (full caption heading also contains the figure number).
- \@figtab@tablelabeltext
  Caption heading text for tables (full caption heading also contains the table number).
- \@figtab@columnseparation
  Horizontal separation between two consecutive columns in a table.
- \@figtab@rulewidth
  Width of rules drawn in a table.
- \@figtab@doubleruleseparation
  Separation between two rules placed next to each other to form a double rule.

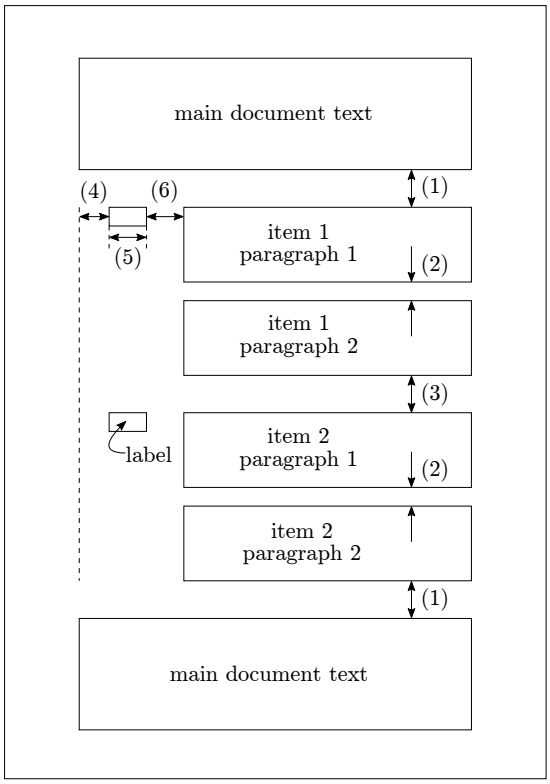

Figure 9.2: Each list consists of multiple entries, called items. Every item has a label which has a box reserved for itself. The textual content of the item is formatted into paragraphs as shown. The dimensions(1) to (6) that control the list layout are described in Section 9.8.

9.8 Lists and enumerations

Macro name prefix: `\@lstenum@`
Lines: 272 – 291

- `\@lstenum@labelmargin`
  Space between left text margin and label box border. Dimension (4) in Figure 9.2.
- `\@lstenum@labelwidth`
  Width of box reserved for label. Dimension (5) in Figure 9.2.
- `\@lstenum@labelseparation`
  Space between label box and item paragraph box borders. Dimension (6) in Figure 9.2.
- `\@lstenum@itemseparation`
  Space between two consecutive items, in addition to item paragraph separation. Dimension (3) minus dimension (2) in Figure 9.2.
- `\@lstenum@paragraphseparation`
  Space between two paragraphs within a single item. Dimension (2) in Figure 9.2.
- `\@lstenum@aboveseparation`
  Space left above and below the list structure in addition to the default paragraph spacing in the main document text. Dimension (1) minus `\parskip` in Figure 9.2.

- `\@lstenum@listinglabeli`
  Symbol used as label for top-level items in `listing` environments.
- `\@lstenum@listinglabelii`
  Symbol used as label for items at nesting depth of 2, in `listing` environments.
- `\@lstenum@listinglabeliii`
  Symbol used as label for items at nesting depth of 3, in `listing` environments.
- `\@lstenum@listinglabeliv`
  Symbol used as label for items at nesting depth of 4, in `listing` environments.
- `\@lstenum@enumerationlabeli`
  Label format for top-level items in `enumeration` environments.
- `\@lstenum@enumerationlabelii`
  Label format for items at nesting depth of 2, in `enumeration` environments.
- `\@lstenum@enumerationlabeliii`
  Label format for items at nesting depth of 3, in `enumeration` environments.
- `\@lstenum@enumerationlabeliv`
  Label format for items at nesting depth of 4, in `enumeration` environments.

9.9 TECHNICAL BLOCKS

Macro name prefix: `\@techblk@`
Lines: 292 – 298

- `\@techblk@headfont`
  Font used in block heading.
- `\@techblk@emphasizedbodyfont`
  Font used in main body of blocks with the `emph` option enabled.
- `\@techblk@normalbodyfont`
  Font used in main body of blocks with the `noemph` option enabled.
- `\@techblk@terminationmark`
  The Q.E.D. symbol to use for blocks with the `term` option enabled.

A

Font selection

Old font selection commands such as `\textbf` or `\textit` are not supported in the `thesis` class. Only the newer font selection commands should be used in the source text. The following sections enlist the supported font selection commands for both, textual, and mathematical modes.

A.1 Textual fonts

Fonts in textual mode are defined by four attributes – size, family, series, and shape. The value of each attribute can be specified by a corresponding set of commands. Sections A.1.1 to A.1.4 list out the commands for each attribute.

A.1.1 Font size commands

The following are the font size commands in decreasing order of size.

- `\Huge`
- `\huge`
- `\LARGE`
- `\Large`
- `\large`
- `\normalsize`
- `\small`
- `\footnotesize`
- `\scriptsize`
- `\tiny`

A.1.2 Font family commands

- `\rmfamily`
  Roman font family.
- `\sffamily`
  Sans serif font family.
- `\ttfamily`
  Typewriter text font family.

A.1.3 Font series commands

- `\mdseries`
  Medium weight font (default appearance).
- `\bfseries`
  Boldface font.

A.1.4 Font shape commands

- `\upshape`
  Upright font shape.
- `\itshape`
  Italic font shape.
- `\slshape`
  Slanted font shape.
- `\scshape`
  Small-caps font shape.

A.2 Mathematical fonts

In mathematical mode, the following font selection commands must be used instead of those presented in Section A.1.

- `\mathrm`
  Roman font.
- `\mathit`
  Italic font.
- `\mathbf`
  Boldface font.
- `\mathsf`
  Sans serif font.
- `\mathtt`
  Typewriter font.
- `\mathcal`
  Calligraphic font (use only for upper case alphabets).

B

Automatic hyperlinks

All references created with the \ref and \eqref commands, citation marks created using \cite (see Appendix C), entries in the table of contents, etc. can automatically be converted into hyperlinks. In order to do this, simply include the hyperref package in the preamble.

```
\usepackage{hyperref}
```

The appearance of the hyperlinks in the document can be controlled using the \hypersetup command. For example, custom colours can be specified for different types of hyperlinks in the following way.

```
\usepackage{xcolors}
\usepackage{hyperref}

\definecolor{hreflink}{RGB}{150,75,0}
\definecolor{hrefcite}{RGB}{200,0,200}
\definecolor{hrefurl}{RGB}{0,50,200}

\hypersetup{
  colorlinks = true,
  linkcolor = hreflink,
  citecolor = hrefcite,
  urlcolor = hrefurl
}
```

In this example, RGB colours are defined using the \definecolor command from the package xcolor. The newly defined colour names are then used in the \hypersetup command.

Detailed information about \hypersetup can be found in the official documentation of the hyperref package.

C

Creating the bibliography

The bibliography can be created in the usual way, using BibTeX. The `biblatex` package must be imported, and a `*.bib` file containing BibTeX entries for each cited reference should be created. This file should then be added as a bibliography resource. For example, if `thesis.bib` is the bibliography resource file, then the following should be put into the preamble code.

```
\usepackage[backend=biber]{biblatex}
\addbibresource{thesis.bib}
```

The `\cite` command can be used to cite any reference defined in the `*.bib` file. Donald Knuth's *The TEXbook* [2] was cited in this document using this command.

The command `\nocite` can be used instead of `\cite` if an entry should be added to the bibliography, without creating a citation mark in the text. For example, Matplotlib has been cited in this document using `\nocite`.

Detailed information on using the `biblatex` package can be found in its official documentation.

Bibliography

[1] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[2] Donald Ervin Knuth and Duane Bibby. *The TEXbook*. Vol. 3. Addison-Wesley Reading, 1984.