On managing large documents

Thomas Thurnherr

Abstract

Whether you are working on a book, a doctoral thesis, or an extensive report, large documents can be difficult. The source code becomes confusing; you scroll back and forth to find what you are looking for; and processing the document can take minutes rather than seconds. In this article, I describe some tools and thoughts which might help to keep large documents organized so you can focus on the essential: the content.

1 Focus on one thing at a time

Focusing on one thing is generally more efficient than trying to do multiple things at the same time. For example, I find it better to work on the document layout, or edit a figure or table, or focus on writing a chapter. It does not mean that I work on the chapter until it is done. Rather, I concentrate on a chapter for a while and then switch to working on the document layout or a figure when I need a break from writing.

2 Keep things separate

A good way to keep focused is to have multiple source files, especially to keep the preamble separate from all contents. To that end, I often create a file `main.tex` with the preamble. In the preamble, I load all required packages, define new macros, set the header and footer, and configure other features of the document layout. Also in the main source file, I load all document contents. This is illustrated in the brief example below.

```
\documentclass{report} % main.tex
% Preamble
% Load packages and set document style
\usepackage{microtype}
\usepackage{biblatex}
\bibliography{references}
\...
% Main document
% Include all content
\begin{document}
        \include{titlepage}
        \include{contents} % toc/lof/lot
        \include{chapter1}
        \include{chapter2}
        \include{...}
        \printbibliography
        \include{appendix}
\end{document}
```

2.1 Include and input macros

In the example above, I used the `\include` macro. The macro loads content from a source file. For example, suppose I physically saved the content of the first chapter in `chapter1.tex`. Now, I can add its content to the document using `\include{chapter1}`. Moreover, `\include` adds a page break before the added content. This is great for chapters, which are physically separated from previous and subsequent contents. To add smaller chunks of content, I can use `\input{filename}` instead, which does not force page breaks. For example, `\input` is a good choice to add a formula or table from a separate source file. Moreover, this command is a great way to reuse code. Finally, the `\input` macro can be nested, which is not possible with `\include`.

Taken together, `\include` and `\input` help to stay on top of things in a large document by splitting the source into multiple files. An additional advantage is that I can reduce processing time of the document while working on it, by using `\includeonly` or by commenting out all `\include` and `\input` macros with content I am not currently editing.

3 Keep the project directory uncluttered

In order to keep the main project directory clean, I usually place figure files in a `figures` sub-folder and chapter source files in a `chapters` sub-folder. This helps to reduce the number of files in the main project directory. To load these files, I now have to provide their path as seen from the main project directory.

```
%Load figure from figures sub-folder
\includegraphics{figures/figure-filename}

%Load chapter from chapters sub-folder
\include{chapters/chapter1}
```

With some additional effort, I can move all meta-files into a `metafiles` sub-folder. Meta-files are files generated by the TEX engine and other programs invoked to generate the output. These have file endings: `log`, `aux`, `toc`, `lof`, `lot`, `bbl`, `bcr`, etc. To save meta-files to a sub-folder, `pdflatex` and other programs need to be called with various options. These are described in their respective manual pages.

4 Use a script for document processing

If you set out to write a lengthy document, it may be a good idea to learn `latexmk` [4]. The idea of `latexmk` was borrowed from `Makefiles`, which are frequently distributed with source code for C programs. A `latexmk Makefile` contains instructions on how to process a document by the TEX engine.

latexmk is distributed with major LaTeX distributions and therefore most likely available on your system. For more information, take a look at the documentation or search for tutorials and example Makefiles online.

Alternatively, with some basic bash or similar knowledge, you could write your own script or (original) Makefile to process the source. The script should at least have two options: 1) to typeset text only; and 2) to properly process references, the bibliography, and glossaries.

5 Mind the package order

Large documents frequently depend on a long list of packages. Sometimes the order in which packages are loaded matters, although this is generally less of a problem nowadays. One notable case remains, however: the hyperref package [2] tends to cause conflicts when used with other packages. As a rule of thumb, hyperref should be last, with some exceptions [6]. Usually, package conflicts are documented in the package documentation or online.

6 Labels and cross-referencing

A label is used to cross-reference a numbered element of a document. The quantity of labels increases with the size of a document and it quickly becomes difficult to remember all label names and to omit duplicates. It is good practice to use a label prefix, such as fig: for figures and tab: for tables, as in fig:workflow. There are no predefined prefixes, but often the first three letters of the command to reference are used. In addition, no prefix is recognized by the TeX program; their usage is entirely for the author's convenience. Finally, I have not seen prefixes used for bibliographic references. Instead, for BibTeX entries, I recommend concatenating the name of the first author, the year, and the first word of the title to form unique citation identifiers.

The showlabels package [5] can help keep track of labels. It prints their names in the margins of the processed document. The package either shows all labels (default) or only labels of specific commands (see example below). To generate the final version of a document, I can either manually remove the package or mute all its functions through its final option.

```
\usepackage[nolabel]{showlabels}
\showlabels{cite}
```

7 Draft mode

The draft option of the document class macro produces a visible mark for Overfull hbox warnings. These are positions in the document where the text or other content reaches into the margins. In addition, the option may alter the behavior of loaded packages. For example, with the draft option set, the graphicx package [1] indicates a figure with a black canvas instead of loading the actual figure. This can drastically reduce document processing time.

```
\documentclass[draft]{report}
```

The ifdraft package [3] allows additional customization of the behavior in draft mode. For example, I might want to define a todo command to mark unfinished content. By defining todo as empty command in final mode, I make sure no todo output is produced in the final version of the document.

```
\usepackage{xcolor}
\usepackage{ifdraft}
\ifdraft{% with draft option
        \newcommand{\todo}[1]{%
                \textcolor{red}{[TODO: #1]}}
}{% with final option
        \newcommand{\todo}[1]{}
}
```

References

[1] graphicx — enhanced support for graphics. https://www.ctan.org/pkg/graphicx. Accessed: 2016-02-25.

[2] hyperref — extensive support for hypertext in LaTeX. https://www.ctan.org/pkg/hyperref. Accessed: 2016-02-25.

[3] ifdraft — detect "draft" and "final" class options. https://www.ctan.org/pkg/ifdraft. Accessed: 2016-02-25.

[4] latexmk — fully automated LaTeX document generation. https://www.ctan.org/pkg/latexmk. Accessed: 2016-02-25.

[5] showlabels — show label commands in the margin. https://www.ctan.org/pkg/showlabels. Accessed: 2016-02-25.

[6] Collection of LaTeX package conflicts. http://www.macfreek.nl/memory/LaTeX_package_conflicts. Accessed: 2016-02-25.

◇ Thomas Thurnherr
thomas.thurnherr (at) gmail dot com
http://texblog.org