Learning LaTeX

Ridho Pratama
Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Bandung, Indonesia
Email: p.ridho@students.itb.ac.id

***Abstract*—While using WYSIWYG ("what you see is what you get") text editor, like Microsoft Word or Google Docs, have you ever tried to format something to what you want, and getting frustated because the result is not what you want, while the editor also messing up the rest of your works? Me too.**

But there are better alternatives to write documents professionally. One of the alternatives is LaTeX. With this paper I would like to learn and use LaTeX to write this paper, and share the experiences in learning to use LaTeX.

I. Introduction

LaTeX (stylized as LaTeX), is a document preparation system as an improvement on TeX system created by Donald Knuth. When writing using LaTeX, the writer uses plain text instead of formatted text found in WYSIWYG text editor like Microsoft Word. The writer uses markup tagging to define the structure of a document, to stylize text in a document. The writer usually writes the document in a .tex file, and then use program to compile the document into output file (such as PDF or DVI).

Documents produced using LaTeX usually are of higher quality than documents produced using WYSIWYG editor. This is because when using LaTeX, the writer only focuses on the contents without caring about the formatting like margins, font and fontsize, styling, etc. Those are handled by LaTeX, so the writer could get high-quality and consistent document.

LaTeX is widely used in academia for publication of scientific documents in many fields.

II. Learning LaTeX

A. Installation

Steps to install LaTeX depends on your operating system.

If you use Windows, you could use MiKTeX (https://miktex.org), which is a Windows distribution of LaTeX that includes additional packages and a TeX editor called TeXworks.

On my system which uses Linux, specifically Arch Linux, where usually TeX Live is used. To install TeX Live in Arch Linux, you just run `pacman -S texlive-most` which already includes the LaTeX binary and additional packages.

You could write .tex document in any plain text editor, even notepad. But for ease, I use Visual Studio Code (https://code.visualstudio.com/) with LaTeX Workshop extension installed. The extension adds features that helps you writing .tex document, like auto-completion, auto-compile, and live document preview.

B. Your first document

It's time to create your first latex document. Create a new file named `first.tex`, and fill it with:

```
\documentclass{article}
\begin{document}
    This is my first latex document
\end{document}
```

Listing 1. first.tex

Then open terminal in your current folder and run `pdflatex first.tex` to compile the document into a PDF file named `first.pdf`. Open the PDF file and you should file the text 'This is my first latex document' in it, and congratulations, you have created your first latex document.

C. Document class

You should see that the document in previous subsection already have a lot of things set up for you, like margin, font and font size, spacing, etc, and you only have to write the text you wanted. That part is set up with **documentclass**{article} part. That command lets you specify to the compiler that you want to create an article. There are many built-in document classes already specified for various uses. Class 'article' is for articles in scientific journals, presentations, etc, 'book' is for real book, 'letter' is for creating letters, 'beamer' for writing presentations, and many other for more specific uses.

D. Document environment

The next part is **begin**{document}...**end**{document} That pair of command specify an 'environment' in which the item inside it have some rules applied to it. In the example we used 'document' environment, which tells the compiler that it is our document and then everything inside the environment will be written in output document. This also means, any normal text outside of the document environment will not be printed. In fact, we will get an error when compiling if we have normal text outside of document environment.

There are many other environment for other specific uses, like 'description', 'list', 'enumerate' and 'itemize' to create various numbered and bulleted lists, 'figure' and 'table' to create figures and tables, 'math' and 'displaymath' to type math, 'array' to create array, 'equation' to type equations, and many other more. We also could create our own custom

enviroment or use enviroments create by other people by using packages.

E. Line and paragraph

Create file `second.tex` and type:

```
\documentclass{article}

\title{This is a Title}
\author{Ridho Pratama}
\date{28 April 2019}

\begin{document}
\maketitle

\section{Section 1}
\subsection{Subsection 1.1}
This is first paragraph.
And this is also in first paragraph.

But this is in second paragrah, and you
can split lines in paragraph in the tex
file without affecting the result.

\subsection{Subsection 1.2}
This is second subsection

\end{document}
```

Listing 2. second.tex

And compile the file. There is many other thing in the listing, but we will focus on section 1.1.

You could see that how the text are typed in the .tex file didn't affect how the paragraph are being splitted in the result file. In latex, texts are usually typed in paragraph. In the text file, you separate between paragraph using blank lines. And so, the lines without blank lines between in will be treated as one paragraph in the result. LaTeX will automatically add separating spaces.

F. Titles and author

Back to listing 2, the code in line 3-5 specifies the documents title, author, and written date. But this only specify the attributes and doesn't create any text in output. To put the title and author in the output file, we use `\maketitle` to print it.

G. Document structure

In listing 2, There are also one more thing, that is the section and subsection. Those are the document structure marker, using that we could structure our document into sections and subsections. The section numberings are automatic. Using the document structure could also help us in generating automatic table of contents.

This is specifically for 'article' document class. For other classes, the structuring might be different, like 'book' class has chapter structure, etc.