LaTeX for the IB

Get a 7 with style.

Licensing and Contributors

OpenTextbooks, Coming Soon™...

Contents

1 What is LaTeX?

2 Installing LaTeX

2.1 Mac

2.2 Windows

2.3 Linux

3 Writing LaTeX

3.1 The Easy Way

3.2 Editors and Other Goodies

4 Fundementals

4.1 Document Structure

4.2 Basic Formatting

4.3 Whitespace and Pages

4.4 Headers, Footers and Page-numbers

4.5 The Titlepage

4.6 Contents, Bibliography and Appendices

5 Writing Maths

5.1 Math-Mode

5.2 Maths Symbols

6 Writing In Foreign Languages

6.1 Spanish, French and Accents

6.2 Asian Languages

6.3 IPA and Miscellaneous

7 Programming and Advanced LaTeX

7.1 Turing Completeness

7.2 Macros

7.3 Writing Packages

7.4 Writing Classes

8 About LaTeX

First of all, let's clear up what LaTeX is and what it is not. It is

- typesetting system,
- programming language focused on macros,
- free

but it is not

- word processor,
- text editor,
- complicated.

In a nutshell, you create a *.tex* file (normal text file with different extension), run a converter and out pops a nicely formatted *.pdf* file. But for the software to know what to do, you need to structure your file appropriately.

Few advantages of LaTeX being mostly text files are that it is lightweight to share and easy to use with version control. And you don't even need graphical interface to edit the files.

9 File structure

In LaTeX you use different commands, based on how would you like the output to look like. To tell it that the command (or macro) has started, the symbol \ is used. And since most macros need some specific amount of text to work on, we use curly braces { and } to delimit that. In general, { and } are used to feed mandatory arguments to functions while [and] are used to pass optional ones. For example, to make text italic, we would use `\textit` command, which would look like the following: `\textit{lorem ipsum}`.

The comment glyph is %. This means that everything from the first % on the line until its end is completely ignored at compiling.

At the beginning, we need to tell the compiler what kind of document will we be working on. We do this with the command `\documentclass`. We need to provide the document class, but we can also specify font size, paper size and a few other things. Some of the more important available classes are

article for scientific articles, reports, documentation ...

report for longer reports split into multiple chapters, smaller books and simmilar

book for actual, proper books

memoir is based on `book` class, but you have a lot of freedom when it comes to changing the output (whether or not is this a good thing is a topic for another time)

letter for writing letters

beamer for creating presentations

For example, this document uses the following setup: `\documentclass[12pt,a4paper]{article}`.

After that, comes the time for including different packages. They usually provide pre-written macros that will either be used later on while writing or will help us format the document the way we like. That is done with `\includepackage` command, where we need to tell what package we want and optionally give it parameters. Few of the most used packages would probably be `inputenc` with `utf8` option, to enable international characters, `geometry` to set up page layout (especially page margins) and, if you are working in another language, `babel`. The example of the first one would be `\includepackage[utf8]{iinputenc}`.

The third part is available for us to change document settings and define our own macros.

And finally comes the content section. It is started with `\begin{document}` and ended with `\end{document}`. Nothing before the former or after the latter will be directly shown. Even more, anything that comes after `\end{document}` is completely ignored; the compiler stops right then and there.

In the main part, you are free to write your content and use macros. However, there are a few things to keep in mind. First of all, LaTeX ignores all whitespace. That means that no matter how many spaces you enter, in the output there will be only one. Any spaces at the beginning/end of the line is completely ignored. Even line breaks are ignored. This is great, because you can structure the original file so that it it easier to read, while not changing the output at all. If you do, however, want a line break, you can force that by using `\\`. To start a new paragraph, just leave one line blank in between. But if you want more space between different parts of your text, forget all of multi-space/multi-new line workarounds. With LaTeX you have commands for that as well.

9.1 Whitespace

To produce additional whitespace, you have a few macros at your disposal, all of which take one argument, that is amount of whitespace to apply. LaTeX can understand the following units by default:

pt point (1 in = 72.27 pt)

pc pica (1 pc = 12 pt)

in inch (1 in = 25.4 mm)

bp big point (1 in = 72 bp)

cm centimetre (1 cm = 10 mm)

mm millimetre

dd didot point (1157 dd = 1238 pt)

cc cicero (1 cc = 12 dd)

sp scaled point (65536 sp = 1 pt)

Two most commonly used spacing macros are `\hspace` and `\vspace` that will insert horizontal and vertical space respectively. If however, a line or page break would occur at a point where either command is used, no whitespace is inserted at all. To force it, append the command with `*` (for example, `\vspace*{20mm}`).

If you wanted to fill larger amounts of space, LaTeX offers `\hfill` and `\vfill` commands. When present, they will try to take up as much space as possible on the current line/page, while still leaving space for the remaining text. There are also two variants of the first command, `\hrulefill` and `\dotfill`, which will foll the space with a rule (line) or dots, instead of whitespace. The examples of all three are:

Lorem ipsum

Dolor______________________________sit

Amet...consectetur

As a matter of fact, they act like springs, which means that where multiple occurences happen on a single line, they'll divide the space as evenly as possible. In the following example we use text, 2x `\hfill`, text, `\hfill`, text and we get

Lorem ipsum dolor

With the free space on the left twice as big as the one on the right.

When it comes to formatting pure spaces, LaTeX has a lot of freedom. For example, if a dot is after lowercase letter (i.e. ending a sentence) there will be a larger space than after a dot after an uppercase letter (i.e. after an initial). If we want to force a small space, for example after `etc.`, `Mr. Smith` and a lot others, we can do so by preceding the space with a `\`, which will look like Mr. Smith as opposed to Mr. Smith. We also have a possibility of inserting even smaller space, by using `\,`. This is used when connecting units to a number, for example 42 ly against 42 ly.

Spaces also tell the compiler where he can make a line break shall the need arise. To avoid that we can use non-breaking space, which is marked by the character ˜ (~). It will look exactly the same, so if we write `Example 7` and `Example~8`, we will see no difference, unless a line break would fall exactly there. Our input would look like: Example 7 and Example 8.