Introduction to LaTeX and to some of its tools

Gianluca Pignalberi, Massimiliano Dominici

Abstract

Writing has a long history. Shorter is the history of typesetting and even shorter is the history of digital typography. Nevertheless, the latter gained an unprecedented importance because of its capability to speed up the process of feeding human being with well-composed information.

Our lessons, of which this is the number zero, are focused on a digital typesetting system that has come to light in the late 70s of 1900. It was intended to typeset scientific books; it is used to typeset nearly everything. It is TeX.

In this short course we will give an overview on how TeX and its most famous macro package LaTeX helps engineers, scientists and professionals to compose their documents, being them books, papers, reports, presentations, posters.

Sommario

La scrittura ha una lunga storia. Più breve è la storia della composizione tipografica e ancor più breve è quella della tipografia digitale. Ciò nonostante, quest'ultima ha guadagnato un'importanza senza precedenti per la sua capacità di velocizzare il processo di rifornire l'essere umano di informazione ben composta.

Le nostre lezioni, di cui questa è la numero zero, sono incentrate su un sistema di composizione digitale venuto alla luce nei tardi anni '70 del 1900. Questo era pensato per comporre libri scientifici; è usato per comporre quasi tutto. È TeX.

In questo breve corso daremo una panoramica di come TeX e il suo più famoso pacchetto di macro LaTeX aiuta gli ingegneri, gli scienziati e i professionisti a comporre i loro documenti, siano essi libri, articoli, relazioni, presentazioni, poster.

Part I: Digital Typography and Not

1 Typesetting Systems *vs* Word Processors

Computers have often been (and currently are) used as typewriters, i.e., to edit text. Text editing has evolved: from the simple words juxtaposition with no hyphenation, monolingual spell check, monospaced font and fixed spacing (between words and lines) and dimensions (for the document) to character kerning, spell and grammar check in different languages, fancy OpenType fonts with contextual shapes and better page and document setup.

Text processing is wonderfully performed by word processors (WPs from now on): Word, Writer and, back in time, DecWrite, WordPerfect, LetterPerfect, WordStar... But, while WordStar and LetterPerfect were not that fancy (they existed when printers were light years far from the Apple LaserWriter), the others listed after Word and Writer were closer to modern WPs. While ancient WPs just allowed text arrangement on the page (no external elements were allowed; no font selection; no fanciness) and spell check, modern WPs are capable of much, much more. They have partially invaded the world of typesetting systems: some minor publishing houses use WPs to create their camera-ready books and journals and we personally set up a LaTeX class for such a publishing house that used to typeset an academic journal in Word. Even a Mid-Pharma company used Word to produce its official reports... until someone discovered that Word was not able to include something like 800 PDF automatically-generated tables into the same report and tried to switch to LaTeX through LyX. Since WPs and typesetting systems have different targets, WPs are not yet as sharp and versatile as typesetting systems are and typesetting systems do not see text as their "core business", i.e., they are not intended as "click-and-type" programs.

Nobody less than insane would pretend to compare the performances of such different tools. WPs are programs mainly intended to process text and to let unskilled users produce reports and other documents with a decent look. Typesetting systems (once named DTP after DeskTop Publishing) are powerful and highly specialized tools to professionally produce newspapers, magazines, books, journals, fliers, banners, you name it. They just process text as one of the zillions tasks they do but have to typeset it the best way as possible. LaTeX *is* a typesetting system and so it is its typesetting engine: TeX. As we will see in the next section, being LaTeX a command line system, it even has to rely on an external text editor. The aforementioned target difference and the LaTeX lack of a built-in text editor turns useless those endless discussions about "Is it LaTeX better than Word?", as in Oetiker *et al.* (2018, pp. 3–4) (that shows pros and cons), Knauff and Nejasmic (2014) and Blanco (2015) (that advise Word) or Bloch (2017) (that explains why LaTeX should be

better). It is pointless to compare two programs that perform different, though partially superimposed, tasks. It is pointless to measure how fast users input text and tables when the hardware is not the same and you do not specify who had the autocorrect activated or not: this is a word processors tool—not typesetting systems'—that might increase the speed performance of users. It is pointless to measure how sharp users input text and tables and not to measure how close to the original is the final document look (but Knauff and Nejasmic (2014) mentions that).

WPs care about the fact that every line in a page is good and well hyphenated, regardless of the page quality; typesetting systems not only care that a line is more than good and correctly typeset and hyphenated but should care of the page quality too. TEX only issues a page when the page is typographically the best possible according to TEX's internal rules. At last, Word does not care of documents back-compatibility, unlike LATEX. More on that topic in section 4.

2 Interactive and Non-Interactive Typesetting Systems

Despite the majority of users just know interactive[1] programs, there are still many programs that are not interactive. Typesetting systems are no exception and probably the most representative non-interactive programs and troff and TEX. Interactive (and visual) typesetting systems are QuarkXPress, Adobe InDesign, Microsoft Publisher, Scribus, and the very ancient (and dismissed) Ventura Publisher and Aldus PageMaker.

The main difference between interactive and non-interactive typesetting systems is that an interactive one shows you in real time the result of your actions and your actions are usually dragging and dropping boxes in a visual interface, while a non-interactive program accepts whatever action you want but shows you a result only when you instruct it to show. So, if you delete a sentence in an InDesign text, you immediately see the modified text; if you do the same with TEX, you will only see the result after compiling the new text and opening the resulting DVI or PDF.

Just to roughly cut the users set, graphic designers consider more productive using interactive programs; programmers think otherwise and it seems to us that programmers are closer to old linotypers, who had to rely on their experience to produce text lines that where not too empty or too full before seeing them cast into lead.

We are now entering the world of an old, yet way too powerful typesetting system: TEX, and its almost universally used "hi-level access gate": LATEX.

3 TEX As a Non-Interactive Typesetting System and a Programming Language

TEX is a typesetting system designed and programmed by Donald Knuth at the Stanford University. Its first release came to light in 1978. In Knuth (1999, chap. 1) the author explains why he decided to write such a program: the first volumes of his masterpiece *The Art of Computer Programming*, first typeset with Monotype, needed to be updated but that technology had been dismissed in the USA. The available technology was unable to get at least a similar result so he decided that a program able to typeset books and a program able to generate the needed fonts had to be written: they where TEX and METAFONT.

TEX is a non-interactive typesetting system, so the users have to instruct it—program it—on how to output the desired document. Once the program is ready, TEX compiles it and—hopefully—outputs the document (in DVI format).

TEX programming language provides the users about 900 commands, tests and so on. It is straightforward to realize that TEX is extremely powerful, yet not much user-friendly in the way we currently intend that friendliness.

The DVI documents needed one more step to be ready for a printing service: a DVI→PostScript conversion had to be performed. The program `dvips` accomplished that task.

Years later pdfTEX started outputting PDF documents.

These programs have been partially superseded by XETEX and LuaTEX: both support TTF/OTF fonts and LuaLATEX adds to TEX a powerful yet simple programming language: Lua.

As you may figure out, TEX is not a program that can be installed by itself, without any companion program or file. Indeed it comes with packages known as *distributions*. Despite several distributions have been available for different operating systems, it is now common to see that three distributions polarized users: MikTEX (`https://miktex.org/`) on Windows systems, TEX Live (`https://www.tug.org/texlive/`) on Linux systems and MacTEX (`https://www.tug.org/mactex/`) on Apple computers. While you can refer to the related websites for instructions on how to install them, you can also refer to Gregorio (2010) for a skilled, though quite outdated, guide to install TEX Live on a Linux system.

1. This concept is often mistaken with visual. It is different because "visual" means that you see fancy interfaces, use the mouse or other similar input devices; "interactive" means that the program immediately reacts to your actions and shows you updated results while you keep working. We remember of an old visual spreadsheet: Borland's Paradox. It had a switch: automatic or manual update. The second did not propagate users modifications to connected cells until an explicit update had been issued. Not that interactive.

4 LaTeX, a Macro Package Built on Top of TeX

TeX is quite a complex typesetting system and it is definitely not user friendly. Its nearly 900 commands can scare both those users who only rely on point-and-click operations and those who are not scared of using command line. Indeed those commands express the complexity of typography and the power of the TeX language (which *is* a programming language and a sort of page-description language).

Since authors are not supposed to know anything of typography—they are not typographers—it would be a bad practice asking them to use such a complex typesetting system to write their manuscripts.[2] Because of that, Leslie Lamport wrote a macro package that, while using TeX as its typesetting engine, had to provide users with a very mnemonic set of commands to structure a document. Those users—authors—had to concentrate on content, not on document look, so they just had to tag text as chapters, sections, emphasized, footnotes, and LaTeX would deploy those elements in the right way.

Going back to the *querelle* LaTeX *vs* Word, the only contact point and consequent reason to compare those tools (but it does not mean that such a comparison is meaningful) is that they both are for authors (concentrate on the content, not on the look!), though it is not immediate to find Word users who use styles. Those users who do not use styles keep on applying properties to text by hand—seldom in a coherent way. They act as typographers more than authors; should we compare Word to TeX or to InDesign and the other visual programs similar to it? We think we should not.

After studying the code of Lamport's macro package, Frank Mittelbach decided to re-program it to make it faster and less demanding when compiling a document. He also leads the LaTeX Project that provided us with LaTeX 3, a huge improvement of the language. But LaTeX is not the only macro package based on TeX. Others are ConTeXt, pdfLaTeX, XeLaTeX and LuaLaTeX (until it lasts).

While pdfLaTeX, XeLaTeX and LuaLaTeX are equivalent to LaTeX because they use at least the same set of commands, ConTeXt uses a completely different macro set. That indicates how flexible TeX is.

5 Why Text Is Better Than Binary?

Nearly every source files in the TeX ecosystem are text files (with the obvious exceptions of the compiler, external images, fonts): user documents, packages and classes, font size files, configuration files and the macro package itself (i.e., LaTeX) are text files.

A text file is a file whose content is stored as a sequence of character data. That content might be not immediately understandable (because written in a foreign language or because carries a hidden meaning) but it is surely human-readable: we recognize something resembling letters, numbers and symbols when we open it with a text editor and do not see strange, unprintable symbols or hear beeps. A binary file, on the contrary, may store its content in a more efficient way (i.e., a sequence of four-digit characters may be converted into a two-bytes integer number) but this way is usually hard to read because we do not know *a priori* the way information is stored. Humans cannot even think to read a binary file with a text editor because they could only see a sequence of strange, often unprintable, characters and symbols and hear beeps here and there (or nothing, if the picked editor does not print the BEL, ASCII symbol n. 7).

Why should a text format be picked instead of a more compact binary format? Here we list text format pros. Somebody else would instead list binary format pros and be right anyway. It depends on the task you need to accomplish.

Since a text file can be edited with whatever text editor, TeX documents can be edited even when we do not have the original editor or the compiler on our computer. It means that a TeX document can be edited with vi,[3] Emacs, Notepad, you name it, not only with a specific editor as those we will see in section 9. It can be edited on machines different than the compiling one, even running different operating systems (OS from now on), and even remote editing via telnet or ssh can be easily performed. Of course, you need TeX to generate the final document, being it PostScript or PDF, but it is unnecessary for editing the source document. Configuring the most part of LaTeX files is as easy for the same reason: the content is clear, though modifications might be a hard task for newbies.

In a professional environment it should be normal to keep track of the changes made to documents. Even better, a team might work on the same documents and it has to be possible to coordinate and integrate the job. These are the typical cases where a version control system has to be adopted. Version control systems have always correctly managed text files; they used to behave worse with binary files: being difficult to store subsequent versions as deltas, the files were entirely saved. Even though Subversion and other similar programs efficiently manage binary files (Rooney, 2005, p. 6

2. It might even be a bad practice asking authors to write manuscripts with Adobe InDesign or Scribus. They see a white page but might find surprising not to see the cursor in the home position on that page to simply start writing. They should guess, or learn, that they can do it only if they put a text box on the page.

3. Some readers may wonder why vi is not capitalized. Please refer to Robbins *et al.* (2008) to see that it is officially lowercased and, by the way, it is not pronounced 'six'.

and chapter 2), TEX immediately took advantage of that facility: system managers could pick the version control tool more suitable to their needs and to the needs of "their" TEX users.

A Unix filter to quickly compare a source file against another one—`diff`—can output a complete list of differences between two text files, while it will only state "differ" in case we compare two binary files. Along with this filter come some other useful tools like `sdiff`, `diff3`, `patch` that you can read about in the corresponding man pages of Unix systems or in books like Powers *et al.* (2002). A more useful tool for LaTeX users is `latexdiff` that compares two, and only two,[4] versions of a LaTeX document. It can output a document that highlights the differences between the compared files.

In conclusion we can say that text format helps TEX users to rapidly and versatilely operate on the most part of the files they work with. As we will fully understand later on, another advantage for users is that they can look at the most part of source code to get some help in writing custom commands.

6 LaTeX File Format: the Healing Text

It seems to be quite rare to find a typesetting system that stores its files as text files instead of binary files. It is understandable because every commercial vendor cares of the way it stores data and keeps them hidden from external eyes (reverse engineering is expressly forbidden...). Indeed Scribus, that stores users documents in XML, is free software. Well, as the contemporary trend indicates, XML is currently used as one of the possible file formats even for proprietary software like Microsoft Word and Adobe InDesign, so going back to a form of textually-stored data is more than a simple hope.

TEX has been on the market for about 40 years, so it has been programmed for a (computer) world that had quite narrow character sets and a wider ecosystem. Even the 8-bit ASCII set, with its 256 symbols, was too narrow to allow multilingual documents without switching between different encodings. LaTeX had quite a smart way to represent Latin extended glyphs without any switch (see section 8.1.2). The conditions are far better now because LaTeX source documents can be encoded in UTF-8, one of the Unicode text encodings and this fact widens the number of potential users and uses. Unicode formats are a kind of enlarged ASCII and the newer editors manage them without problems, provided your computer has Unicode-compliant fonts installed.

Due to the way Unicode encodes its characters, it may be possible to determine whether a file has been corrupted somehow: some mono- and multibyte sequences are not legal and cannot address to any glyph. You can read about this in Giacomelli and Pignalberi (2018) and the related bibliography. A nice and quick method to validate a UTF-8 file has been suggested in `https://stackoverflow.com/questions/115210/how-to-check-whether-a-file-is-valid-utf-8`. It is trickier to detect a corruption that transformed a legal glyph into another legal glyph. Of course, despite errors correction is not as straightforward as errors detection, some attempts to restore a corrupted text can be done: the first step is fixing the reserved part of the mono- or the multibyte character that is not legal, in the hope that the remainder part has not been corrupted itself; then a spell checker might help in case the "restored" character is in a word and this word is easy to correct; in the end, a visual inspection might be useful.

4. No equivalent of `diff3` seems to exist.

Part II: Understanding a LaTeX Document

Bonus Section: Compiling a LaTeX Document

Before discussing in detail how to write a LaTeX document, we should understand how to compile it, i.e., how to get a DVI or a PDF out of the source document. Though we will see some friendly tools in section 9 and 10, we see now the hard way.

Linux and Mac OS X users have to open a terminal; Windows users—a command interpreter. After "traveling" to the directory containing the document to be compiled (let us suppose its name is `document-name.tex`), write

```
latex document-name
```

Once the compilation successfully ends, a DVI document has been generated. This has to be post-processed with `dvips` to get a PostScript file. You might prefer to directly generate a PDF, so you are likely to use `pdflatex`, `xelatex` or `lualatex`.

For those who do not love the command line it may be easier to put the compiler icon onto the desktop and drag and drop the document icon onto the compiler icon, should the window manager support those operations.

7 The Structure of a LaTeX Document (part I)

A LaTeX document, that we know being a text file,[5] contains the complete content along with the

5. As we'll see in section 8.3, a LaTeX document can be subdivided in more than one text file, provided the structure coherence is preserved.

information necessary for LaTeX to typeset it. The document is composed by 1) a preliminary part of code—the preamble—containing the general typesetting rules, in the form of a class file, along with the additional commands we can use while typesetting the document and 2) by the document content—the main body or document body.

We talked about commands without introducing them. A LaTeX command is a character sequence opening with a backslash (\) followed by a word or a special symbol. It is interpreted in a special way by the typesetter. It may have 0 or more mandatory arguments along with optional arguments. When invoking the command we have to enclose the optional parameters, if any, in brackets before passing the mandatory parameters individually enclosed in braces if they are 1 or more.[6] Not all the commands are accessible to users. Part of them have been written only to be used by other commands. Commands will be a main topic along this and the subsequent lessons.

The preamble starts with the `\documentclass` command and ends when the main body starts. `\documentclass` specifies which class to use to typeset the document. The preamble is completed by using additional packages, custom commands, needed controls. In these ways we can both fine tune the general typesetting rules of the class and add unforeseen commands to accomplish tasks that are specific to our document.

The main body, enclosed in the pair of commands `\begin{document}`-`\end{document}` (TEXnically, enclosed in an *environment*), contains the document content in the form of text, command+text, environment+text, command. It sounds odd; be patient until section 8. Don't forget to notice that everything written after `\end{document}` will be neglected by LaTeX.

Since we're talking about a program that is also a programming language, we can't forget to mention that we can add comments to our source files. A comment in LaTeX starts with a percent sign (`%`) and lasts until the end of line.

7.1 Preamble analysis: document classes

As it is already clear, the first command in a LaTeX preamble declares which class the document is typeset according to.

The document class, which is stored in a file with extension `cls`, is a kind of configuration file containing the TEX code and the user commands necessary to typeset a specific type of documents. The class adds code, or substitutes part of it, to the macro package (i.e., LaTeX). It is mandatory to specify the class we use: without it LaTeX would not know what kind of document it has to typeset and what look it has to have.

LaTeX provides a handful of classes of general use: book, report, article, letter, slides. The Comprehensive TEX Archive Network (CTAN from now on; `www.ctan.org`) provides users with classes for nearly every need.

Of course, a document has to be "tailored" for the class of interest: a document initially intended to be a book can be hardly transformed in an article without restructuring it because, for instance, a book has chapters and an article has not. But it should be straightforward to compile a document initially intended to be a book into a report or a book typeset with a custom class of a specific publisher.

Let us now see a couple of examples, mutually exclusive, of `\documentclass` possible uses:

```
\documentclass{standalone}

\documentclass[a4paper,11pt]{article}
```

The first one instructs LaTeX to use a class specifically designed to arrange unstructured text or figures in the page and to crop the result. The second one tells LaTeX to typeset an article in A4-page format and with 11 points[7] text font. Examples showing documents resulting when using the above `\documentclass` instances are presented in section 8 (figures 1 and 2).

7.2 Preamble Analysis: What Is a Package?

As already mentioned, a LaTeX document could include some packages to use. The command that includes a package into a document is `\usepackage`. A package is historically a file with the `.sty` extension. Despite users can decide to store packages in files with different extension, `\usepackage` will only consider—and look for—files with `.sty` extension.

A package is a (not necessarily) small piece of code containing one or more brand new macros, one or more pre-existing but modified macros, one or more declarations. Substantially, a package is an injection of code to LaTeX so that it may rely on new features and/or on pre-existing but customized features.

Users can be asking themselves: "which is the difference between a class and a package? They seem to do the same things." It was actually true with LaTeX 2.09 (better, there were not classes at all). Since LaTeX 2_ε we may say that while a class instructs LaTeX to manage a whole document, a package is a tool to fine-tune or enhance the current behavior of the pair macro package-class.

6. Well, it is a little more complex than this: if a parameter has more than one character, we have to group it, i.e., surround it with braces. Otherwise, only the first letter will be considered the actual parameter passed to the command.

7. LaTeX main classes only let you pick 10, 11 or 12 points as the main text size. Other sizes are judged unsuitable for books and papers. But you are free to write a class and the corresponding `.clo` file to meet your needs.

A package is not characterized by it size: we have nearly one-liner packages (i.e., indentfirst) and 1 MB packages such as xq. Anyway, since LaTeX 2ε has been issued, the package is not a piece of software meaningful by itself: you have to use it in a more general context of a document using a class.

Also packages may have options that you declare as optional arguments to `\usepackage`. The manuals of the packages we intend to use will instruct us on the arguments we may set up.

7.3 Preamble Analysis: Completing the Basic Information

When dealing with LaTeX documents, we can be presented a huge series of text files. Even though currently we can be almost sure that a recent text file has been encoded in UTF-8, we cannot be as sure when managing older source files. Despite LaTeX can manage different encodings, it wants us to declare the source encoding. Not only the encoding tells how glyphs are arranged (what code number corresponds to what glyph), but also tells what OS has been used to edit the source file.[8] The package to use to specify the input encoding is inputenc. The actual encoding will be passed as the optional parameter, e.g., `\usepackage[utf8]{inputenc}`. This only applies to LaTeX and pdfLaTeX. XeLaTeX and LuaLaTeX assume that the input file is UTF-8 encoded.

When using LaTeX and pdfLaTeX we also have to specify the font encoding. Roughly speaking, the font encoding tells LaTeX which group of glyphs it has to use while typesetting the document. The package needed for such a specification is fontenc and the T1 encoding usually suffices. We will pass T1 as the optional parameter to `\usepackage` when loading fontenc. You can read the whole story in Mittelbach *et al.* (2016).

The default language used to hyphenate LaTeX documents is English. There are two packages that let us load additional or substituting languages and select them at our need: babel (normally used with LaTeX and pdfLaTeX) and polyglossia (especially programmed for XeLaTeX and LuaLaTeX). Their usage is not uniform: for instance, if you listed the languages as packages optional parameters:

```
\usepackage[english,italian]{babel}

\usepackage[english,italian]{polyglossia}
```

you would obtain Italian as the main language with babel and English as the main language set up by polyglossia. Unfortunately polyglossia no longer supports this technique since version 1.2. So, while the babel example is still valid, the right polyglossia example is:

```
\usepackage{polyglossia}
\setmainlanguage{italian}
\setotherlanguage{english}
```

or, in case of more than one secondary language,

```
\usepackage{polyglossia}
\setmainlanguage{italian}
\setotherlanguages{english, french}
```

More details in Charette (2015).

The default font used to typeset LaTeX documents is Knuth's Computer Modern (Latin Modern for XeLaTeX and LuaLaTeX). We may use another font indicating it in the preamble. While XeLaTeX and LuaLaTeX can easily use every TrueType or OpenType fonts, (pdf)LaTeX can only rely on those fonts specifically prepared for it. The best way to load the picked font is to include one of the packages especially written for that task. The presence of such a package in our systems is not a guarantee that the font file is really installed along with the package. Indeed some packages are distributed even though the font is not freely distributed. An example of such a case is URW Garamond.

Setting up the document main font to, e.g., Linux Libertine is as easy as including a package: `\usepackage{libertine}`. The package takes care of all the operations needed to use such a font. The manuals of such packages help us to set up the fonts according to our needs, though not every fonts have the same settings. Some of them allow us to use old style figures instead of lining figures, to allow some special ligatures and so on.

You can refer to LaTeX 3 Project Team (2005) for more information about fonts in LaTeX.

If we decide to use XeLaTeX or LuaLaTeX, we have a better and uniform control on how we set up the fonts. There is a unique package to load: fontspec, which provides us with an interface to load the fonts we prefer and decide which serif, sans serif and monospaced fonts we will use to typeset a document. The commands for doing that are `\setmainfont`, `\setsansfont` and `\setmonofont`; they ask for at least a mandatory argument that accepts the font name as your OS shows it (optional arguments can be used, for instance, to fine-tune OTF properties). It also allows us to link a specific font to a specific language in case we use polyglossia to set up the languages. You can refer to Robertson and Hosny (2017) to get the insights of what we only touched on and we will use in section 8.1.13.

8. The main difference between files edited with different OSs is in the character representing the end-of-line. DOS/Windows use CR+LF (carriage return and line feed, ASCII codes 13 and 10), Mac OS until version 9 used CR, Linux and other Unix flavors use LF. In our case, we do not need to specify what OS generated the document but we have to specify what OS we are currently using to compile it. Indeed encoding files just list how and where some special characters have been encoded depending on the OS and using a different encoding results in putting strange and unwanted glyphs in our final document.

Hello, world!

FIGURE 1: The resulting PDF of the *Hello world* example.

The document preamble will also contain some data needed by some of the document kinds: title, author and date. You just need to issue the commands `\title`, `\author` and `\date` before the document main body begins. They are not automatically added to your pdf; you need to explicitly invoke the command `\maketitle` in the main body exact point you want those data to appear. Every class has a standard way to show those data. Other classes can delete or rename some of those data commands and even integrate them with other data.

European users should note that the American space after a full stop (aka period) is slightly wider than the average space between words in the same line. Since our typographic tradition is unlike the American one, we should remember to issue the `\frenchspacing` command.

8 The Structure of a LaTeX Document (part II)

8.1 Main Body Analysis: Commands for Text

How do we write a LaTeX text? Before answering this fundamental question, we should at last see some examples that show what we discussed up to now and we will start with the usual *Hello world* example:

```
\documentclass{standalone}
\begin{document}
Hello, world!
\end{document}
```

As you may figure out, the preamble here is the only line `\documentclass{standalone}` (a command with a mandatory argument) while the main body is formed by the only line `Hello, world!`. This line only contains text and, quite comprehensibly, everything we write and do not enclose in a special command will be typeset with the main font and justified. No commands or environments have been issued so far in the document body. Let us see how the document appears, once compiled with LaTeX. That is what figure 1 shows.

Let us now change the preamble to this: `\documentclass[a4paper,11pt]{article}`. This time we passed the command a mandatory and two optional parameters. The body remains untouched. You can see the result in figure 2.

Can you see the differences? The first example generates a PDF cropped to show a tiny white frame around the text, while the second generates a PDF with the text in a specific position of an A4 page.

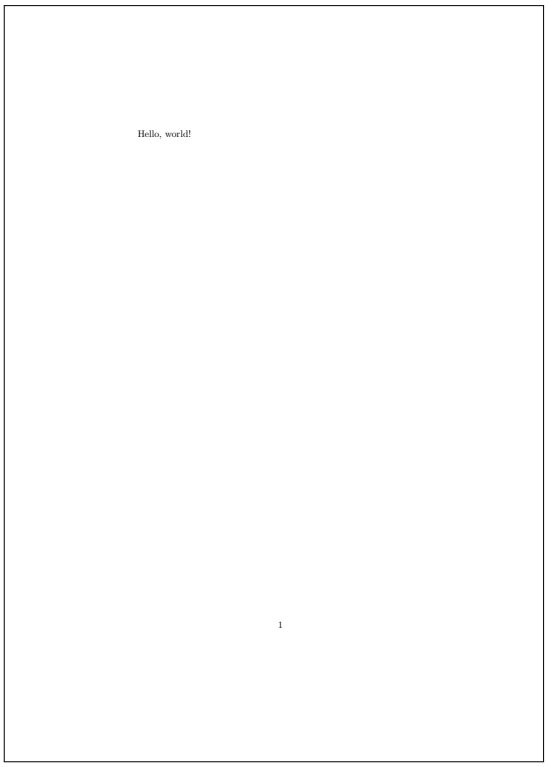


FIGURE 2: The resulting PDF of the *Hello world* example with a new preamble.

The first example puts no other elements in the final PDF, while the second shows a page number. So the document class not only tells us about the look of the document related to the structure but also tells us about the document arrangement in the physical page.

8.1.1 Spaces

WPs users know, but maybe do not notice it a lot, that when they press the space bar the editor cursor shifts to right. The more they press the space bar, the farther the cursor shifts. If they ask the WP to show hidden symbols, they see a number of central dots corresponding to the times they pressed the space bar. A text with more than one space between two words shows inaccuracy.[9] That is why LaTeX disregards the spaces. Or, better, it considers more consecutive spaces as a single space.

In case a user needs to force an additional space for whatever reasons, (s)he can use the command `\␣` (i.e., a space after a backslash). The typical case is after commands like `\LaTeX` that eat spaces following the command.

Other special spaces users should know and use are the nonbreakable space (`~`), that should be used when two words are not meant to be placed in different lines, and the short (unbreakable) space

9. Again, we are simplifying: space between words is somehow elastic because it can stretch or shrink a little bit. It is not a fixed space. Anyway, elastic or not, two spaces are wider than a single space, and that is evident.

(`\,`) that is shorter than the normal space and may be useful in case a text line is slightly longer than you meant or when you represent numbers with their measure unit (and are not using the siunitx package).

WPs users know that they start a new paragraph hitting return at the end of a paragraph. This is false in LaTeX, where a new paragraph starts only after a blank line. A paragraph is usually indented but in special cases or with special classes. How can we avoid to indent a paragraph? As usual in LaTeX, there are several ways and they reflect the semantic intended by the author. If the author meant not to indent a real new paragraph for whatever reason, the right way is issuing `\noindent` at the beginning of the paragraph the author does not want to indent. If the author just wants to interrupt the flow of a line to make the subsequent text to stand (just line an inline example) and this text is part of the paragraph, the way is issuing `\\` or `\newline` where you he wants to terminate a line and do not leave a blank line. The new line is not in a new paragraph according to the LaTeX definition of new paragraph and just mimics it. As you can see, both ways correctly reflect the semantic of the structure.

Similarly to `\newline`, `\newpage` is useful to start a new page.

8.1.2 Special Commands for Diacritic Marks and Special Character

Even though we do not notice it when we type letters with accents and diacritic marks, possibly with a Unicode editor, TeX actually interprets commands. The original way to get an 'a with acute accent' is `\'a`. Can you recognize the command? Right: it is `\'`. The subsequent letter `a` is the mandatory argument. Let us now summarize (table 1) the old way to get diacritic marks and special characters with LaTeX, useful when your keyboard cannot type them directly.

Other special characters easy to get with LaTeX are dashes and quotation marks: `-` is a dash (-), `--` is an en-dash (–), `---` is an em-dash (—); ``` `` ``` is “, `''` is ”, `` ` `` is ‘, `'` is ’, `<<` is «, `>>` is » (these latest quotation marks, named *guillemets* or *chevrons* in French, *virgolette caporali* in Italian, were not allowed with older font encodings like OT1). To obtain an ellipsis character is not as easy, since we cannot type three consecutive periods; we have to use the `\ldots` command (...).

Some other special characters that LaTeX uses, even when we do not tell him to, are the ligatures. The most common are ff, fi, fl, ffi, ffl. There are cases, like in the word *shelfful*, where the 'ff' ligature is not suitable. To avoid it we have either to write `shelf\mbox{}ful` or to group an f (`shelf{f}ul`) to get shelfful.

Two glyphs are especially able to get WPs users confused: the degree symbol and the circumflex accent (^) that we have already seen. The first symbol, that older manuals advice to write as `$^\circ$` (°), can be directly input with nearly any keyboard (°) or obtained via the command `\textdegree` provided by textcomp.[10] Users often mistake that symbol for the superscript o, which is ᵒ (`\textsuperscript{o}`)

Table 1: LaTeX commands to typeset diacritic marks and special characters.

Command	Example	Result
`\``	`\`a`	à
`\'`	`\'a`	á
`\^`	`\^o`	ô
`\v`	`\v o`	ǒ
`\~`	`\~n`	ñ
`\c`	`\c c`	ç
`\b`	`\b o`	o̲
`\=`	`\=o`	ō
`\u`	`\u a`	ă
`\.`	`\.o`	ȯ
`\d`	`\d o`	ọ
`\"`	`\"o`	ö
`\H`	`\H o`	ő
`\k`	`\k o`	ǫ
`\t`	`\t oo`	o͡o
`\oe`		œ
`\OE`		Œ
`\ae`		æ
`\AE`		Æ
`\aa`		å
`\AA`		Å
`\ss`		ß
`\o`		ø
`\O`		Ø
`\l`		ł
`\L`		Ł
`\i`		ı
`\j`		ȷ
`!``		¡
`?``		¿

Do not forget the space in case the command is \ + letter instead of \ + mark.

With very old LaTeX versions you had to put accents on the dotless i or j, otherwise the accent would be put on the dotted characters. With more recent versions it is safe to write `\'i`: you will get the correct accented dotless letter.

It is possible to have the diacritic mark named comma below (ŗ) and its counterpart inverted comma above (ġ) in pdfLaTeX using the *cedilla* command (`\c`), but only with a selected set of letters. To get the comma below under s and t you should keep using the package combelow to get such marks.

10. The textcomp package has no manual, but you can find its symbols in Pakin (2017).

or º (`\textordmasculine`), when used to indicate an ordinal number. The second glyph—ª—is often mistaken for a superscript a, which is ª (`\textsuperscript{a}`) or ª (`\textordfeminine`), to indicate a feminine ordinal.

Before we go on with other special characters, we invite you not to forget that LaTeX has been built on TeX and that TeX is also a programming language. Every programming language has some reserved words and some characters with special meanings. We will keep on talking of this topic in section 8.1.16.

8.1.3 Altering the Text Look

We now may want to change the look of some parts of the texts. WPs users can easily turn text (usually set up in upright serif) into italics, bold or underlined. Quite trickier for them is slanting text or turning it into small caps. So, they can (quite) easily change font shape or series. Stated that underlined is almost banned in typography, LaTeX provides us with easy commands to change the font shape or series (bold, italics, small caps, slanted). The very mnemonic commands to switch shape for a limited portion of text—that will be passed as a mandatory parameter—are:

`\textup` applies upright to the mandatory argument content;

`\textbf` applies bold to the mandatory argument content;

`\textit` applies italics to the mandatory argument content;

`\textsc` applies small caps to the mandatory argument content;

`\textsl` slants the mandatory argument content;

Please notice that these commands only work if you have all the shapes installed in your computer. Let us see a document example:

```
This is roman text.

\textbf{This is bold text.}

\textit Is this italics text?
$\leftarrow$ ???

\textit{This is italics text.}
$\leftarrow$ !!!

\textsc{This is small caps text.}

\textsl{This is slanted text.}
```

This is roman text.
This is bold text.
Is this italics text? ← ???
This is italics text. ← !!!
This is small caps text.
This is slanted text.

Figure 3: Some modified text in LaTeX.

Its result is shown in figure 3. Can you explain the strange result of the third line? No? Do not worry and go reread footnote 6. If it is still unclear, the reason is that `\textit` (and similar commands) looks for a mandatory parameter. Is it our parameter longer than a character? We must surround it with braces. If we do not do that, only the first letter different from space[11] is considered the actual parameter and that is why the third line shows only the I italicized. Anyway, the important thing to understand is that the modification only applies to a defined portion of text or, to use a term common in computer science, the command has a limited scope.

Another widely used command is `\emph`: it emphasizes the mandatory argument content. The difference between it and `\textit` is that the latter always italicizes the text it is applied to while the former italicizes the upright text but uprights text in italics:
`\textit{\textit{Italics} text}` ⇒ *Italics text*
`\textit{\emph{Italics}? text}` ⇒ Italics*? text*

If we want or are forced to switch the shape or the series indefinitely, we can use these other commands: `\upshape`, `\bfseries` (`\mdseries`[12] to re-

11. That is why we told you not to forget spaces in some cases listed in table 1.

12. According to Schmidt (2006), `\mdseries` selects the regular stroke width, unlike `\bfseries` that selects the bold

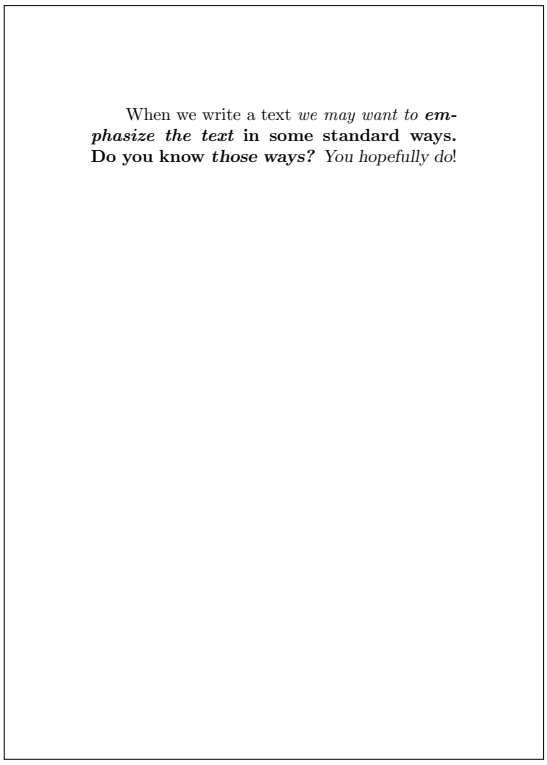


FIGURE 4: This text has been emphasized with commands without arguments.

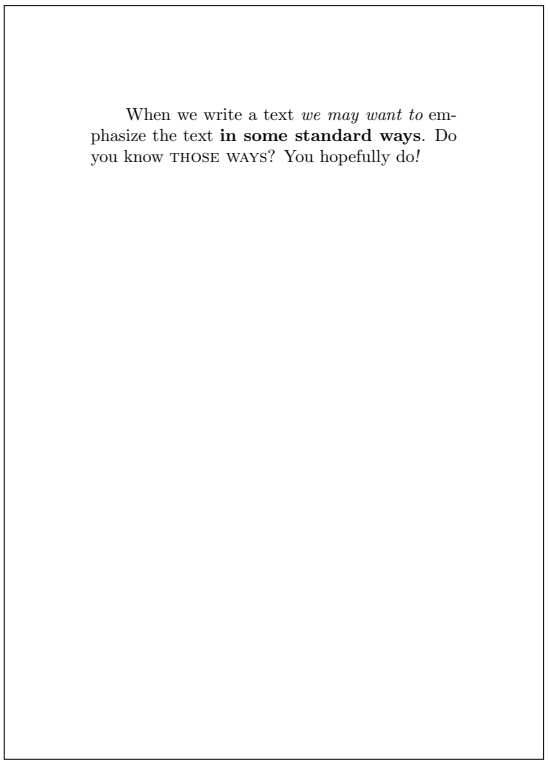


FIGURE 5: This document, apparently similar to that in figure 4, has been obtained limiting the scope of commands like `\itshape` and `\bfseries`. In this case there is no need to explicitly switch to upright roman from another shape or series.

vert it), `\itshape`, `\scshape`, `\slshape`. No argument required. The text after the issue of one such commands will be typeset according to the new shape and series until another similar command is issued. Beware the fact that a `\xxshape` adds up to a `\xxseries`, so if we issue an `\itshape` after a `\bfseries` we get the subsequent text in bold italics. On the contrary, every `\xxshape` or `\xxseries` substitutes the effects of the previously issued corresponding command. In this case, the command scope is not limited. What do you expect from the following example?

```
When we write a text
\itshape we may want to
\bfseries emphasize the text
\scshape in some standard ways.
\upshape Do you know
\slshape those ways?
\mdseries You hopefully do
\upshape!
```

The answer is in figure 4.

If you are wondering if and how is it possible to limit the scope of such commands so that you can avoid to explicitly switch back and forth, the answer is "Yes. It is possible. Just include the switching command and the text it has to be applied to into a group, i.e., include it all in braces." The following example

```
When we write a text
{\itshape we may want to}
emphasize the text
{\bfseries in some standard ways}.
Do you know {\scshape those ways}?
You hopefully do{\slshape!}
```

allows the result shown in figure 5. We see that the text outside groups is upright roman because the scope of the switches we used has been limited by groups.

8.1.4 Altering the Text Font

WPs users can also easily change font family, e.g., from Times to Garamond. Since such an operation usually leads to look incoherence, LaTeX lets you switch fonts in a very tricky way, but lets you easily switch from roman (serif) to sans serif or monospaced font (the TEX refers to as *typewriter typeface* or *teletype*; as we will soon see, abbreviated with tt):

`\textrm` typesets the mandatory argument with the default serif font;

`\textsf` typesets the mandatory argument with the default sans serif font;

stroke width. No way of selecting intermediate, thicker or thinner is allowed. But, of course, LaTeX is not exactly a graphic design tool.

`\texttt` typesets the mandatory argument with the default monospaced font.

The corresponding commands with infinite scope are `\rmfamily`, `\sffamily` and `\ttfamily`.

Since LaTeX stresses typographical beauty, packages that load the main roman font usually also load default sans serif and teletype fonts, both picked to blend with the main font. In some cases (e.g., Concrete) the package even sets up a mathematical font (e.g., Euler). Even though this strategy seems to limit users' "creativity", it avoids typographical blasphemy, since authors should not mind how a document looks: graphic designers have already minded it. You can refer to LaTeX 3 Project Team (2005) for more information about fonts in LaTeX.

As we have seen in section 7.3, in XeLaTeX and LuaLaTeX you are free to associate whatever font to whatever family and switch to one another in the usual way, provided those fonts are installed on your computer. Beware the blend!

8.1.5 Changing the Text Shape in the Page

WPs users can easily change the way a text is arranged in the page. They use to start a document with the text left-aligned and have buttons to change the alignment.

Being a typesetting system mainly oriented to documents for journals, reports and books, LaTeX typesets the main text justified. Of course it is possible to change the alignment and we have two ways to accomplish that: with environments and with commands. As usual, the difference is that the environment clearly shows the scope of the change while the command affects the text from the issue point onwards.

The environments are `center`, `flushleft` (to left align) and `flushright` (to right align); the commands are `\centering`, `\raggedright` (to left align) and `\raggedleft` (to right align).

The following code, using environments, outputs the result shown in figure 6:

```
This is the first paragraph
of this important and significant
text. The \LaTeX\ default behavior
is to justify it.
\bigskip

\begin{center}
This is the second paragraph
of this important and significant
text. We had to use a command
to centering it in the page.
\end{center}
\bigskip

\begin{flushleft}
This is the third paragraph
of this important and significant
text. We had to use a different
command to left align it.
\end{flushleft}
\bigskip

\begin{flushright}
This is the fourth paragraph
of this important and significant
text. We had to use a different
command to right align it.
\end{flushright}
\bigskip

This fifth paragraph is normally
typeset: justified. It's out of
the scope of every environment
previously used.
```

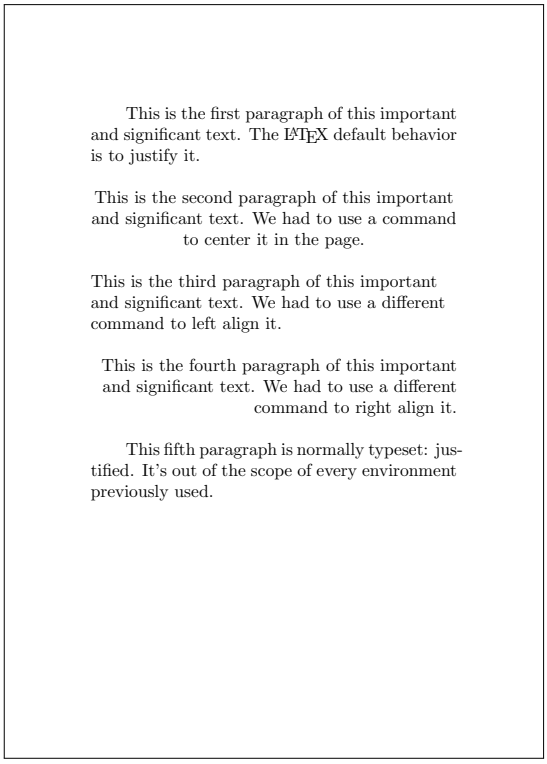


Figure 6: Text aligning with environments.

The following code uses, instead, the commands. The result is in figure 7.

```
This is the first paragraph
of this important and significant
text. The \LaTeX\ default behavior
is to justify it.
\bigskip

\centering
This is the second paragraph
of this important and significant
text. We had to use a command
```

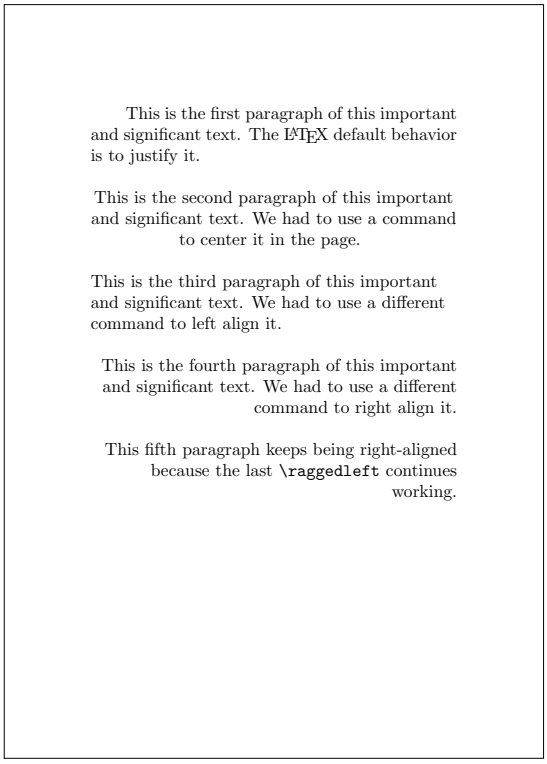


FIGURE 7: Text aligning with commands.

```
to center it in the page.
\bigskip

\raggedright
This is the third paragraph
of this important and significant
text. We had to use a different
command to left align it.
\bigskip

\raggedleft
This is the fourth paragraph
of this important and significant
text. We had to use a different
command to right align it.

\bigskip

This fifth paragraph keeps being
right-aligned because the last
\verb|\raggedleft| continues working.
```

Please, notice the blank line between the `\raggedleft`ed paragraph and the subsequent `\bigskip`. Without the blank line the command is considered part of the paragraph and the last line will end with a space, which makes it look like not being right aligned.

8.1.6 Changing the page format

Despite authors should not manage things that graphic designers should, and despite LaTeX makes not that easy to change page sizes, the geometry package (UMEKI, 2010) has been programmed (especially for class programmers) to make it easier to modify whatever size you want in the page.

8.1.7 Lists

Three are the environments that LaTeX provides to write lists: itemize for bulleted lists, enumerate for numbered lists, description for labeled lists. Each list item must begin with `\item`. This command can get an optional parameter, used to replace numbers and bullets but surely fundamental to indicate labels in a labeled list.

Lists can be nested, regardless of the type but, by default, LaTeX does not allow more than three levels of depth.

8.1.8 Quoted Text, Poetry and Source Code

When writing books, reports, theses, we might want to quote some external contributions in the form of text excerpts. LaTeX offers two environments: quote and quotation. The first one is useful for quoting single-paragraph texts; the second one is intended for multi-paragraph texts, as it indents paragraphs, unlike quote, and the vertical space between paragraphs is equal to the vertical space between lines in a paragraph (`\baselineskip`). The text written into one of these environments (i.e., between `\begin{quote}` and `\end{quote}` or `\begin{quotation}` and `\end{quotation}`) will be quoted according to the specific style. Figures 8 and 9 show the differences.

We might even want to write poetry, where we must exactly control the interruption points. LaTeX has the suitable environment: verse.

```
When we need to write some poetry
in our precious book, we must
exactly control the interruption
points. The \textsf{verse}
environment has been written
for this case:

\begin{verse}
Stick Boy liked Match Girl,\\
He liked her a lot.\\
He liked her cute figure,\\
he thought she was hot.

But could a flame ever burn\\
for a match and a stick?\\
It did quite literally;\\
he burned up quick.
\end{verse}
```

The result is shown in figure 10.

If we need to represent some source code or a similar text, we will use the verbatim environment that typesets the text within it exactly as it is, without any interpretation.

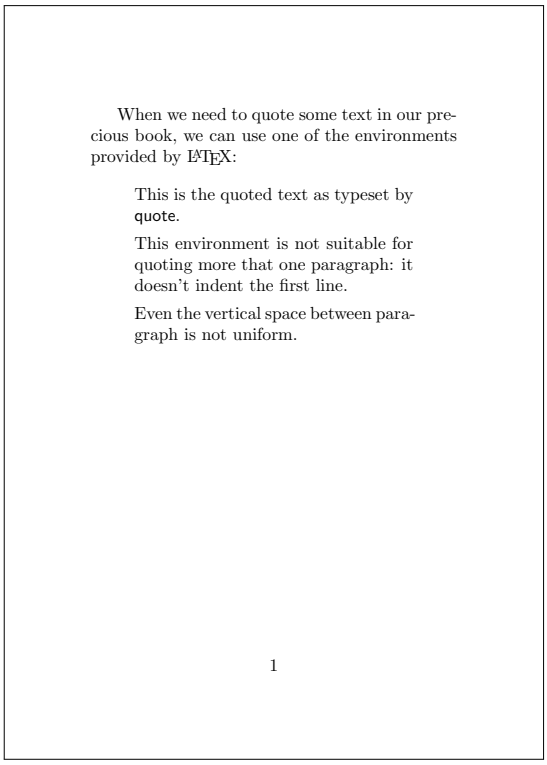


FIGURE 8: This is a multi-paragraph text quoted with quote.

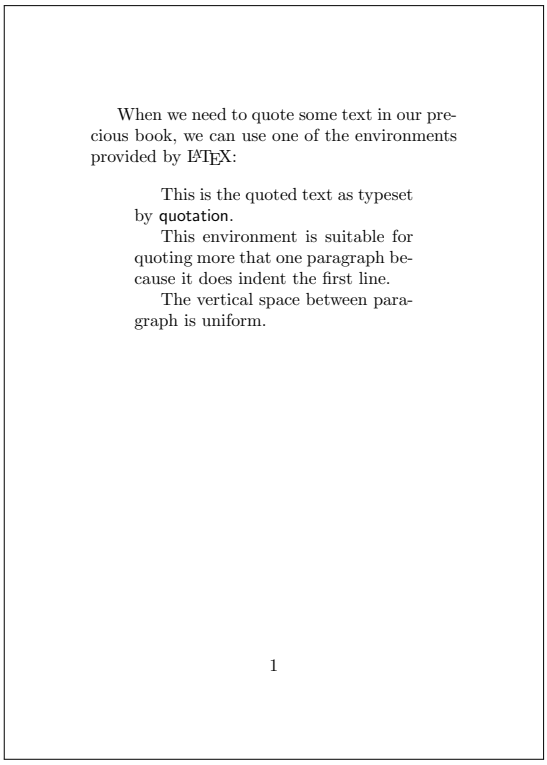


FIGURE 9: This is a multi-paragraph text quoted with quotation.

Pay attention: you cannot nest two verbatim environments, unlike you can do with quote, quotation or verse. That is to say that you cannot show a source code containing the pair `\begin{verbatim}`-`\end{verbatim}` in a verbatim environment. That is why we chose not to include this example source code. Watch a part of it in figure 11.

If we need some verbatim terms along with the text instead of a display, we can use the command `\verb`. It works differently than usual commands: the mandatory argument will not be enclosed in braces but surrounded with a symbol (the same) that will not appear in the verbatim code (for instance, a plus sign—+—or a vertical bar—|—): `\verb+\textit{\LaTeX}+` ⇒ `\textit{\LaTeX}`

8.1.9 Footnotes

Unless you are a novel writer, with the exception of David Foster Wallace,[13] you probably need footnotes in your documents. LaTeX has a specific command, quite surprisingly named `\footnote`. You just need to issue it in the very place you want the footnote mark to appear. LaTeX takes care of correctly typesetting the footnote text (the mandatory parameter), along with the note number, at the bottom of the page.

In some special cases such as footnotes in tables, your footnotes are likely to disappear into thin air. To fix this problem you can use a pair of commands that LaTeX defines: `\footnotemark` and `\footnotetext`. The first one is intended to make the note number or symbol appear in the text; the second one is meant to make the corresponding footnote appear at the bottom of the page. Pay attention that `\footnotemark` may appear in tables, titles and other special cases but `\footnotetext` must be issued out of those structures. Moreover, you have to manually manage the footnote counter if you issues more than one `\footnotemark` before issuing the corresponding `\footnotetext`.

Despite the default footnotes look is defined in the macro package and possibly improved in classes, there are plenty of packages to modify it: footnote, footmisx, bigfoot, footmisc, just to name a few.

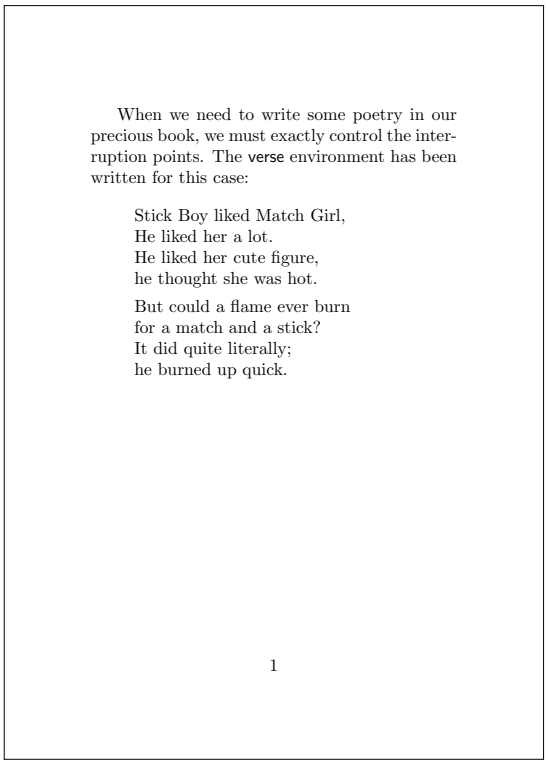
Authors may have reasons for having the footnotes grouped at the end of the documents (endnotes). While LaTeX does not provide such a mechanism, those authors have the endnotes package (LAVAGNINO, 2003) that suits their needs, though visiting CTAN may reserve other pleasing discoveries.

8.1.10 Custom Commands and Environments

In some cases we could find useful to have some custom commands to avoid extensively writing long and repetitive strings of words or to typeset specific portions of text effortlessly.

Custom environments help us define specific styles and rules for needed elements.

13. David Foster Wallace's *Infinite Jest* is crowded of end notes: about 100 pages in 1200 pages of novel. It even has footnotes to some end notes.

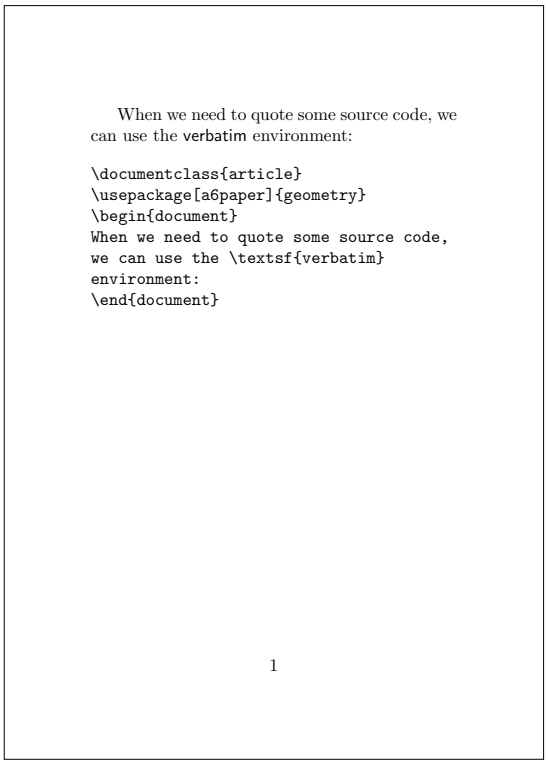

When we need to write some poetry in our precious book, we must exactly control the interruption points. The verse environment has been written for this case:

Stick Boy liked Match Girl,
He liked her a lot.
He liked her cute figure,
he thought she was hot.

But could a flame ever burn
for a match and a stick?
It did quite literally;
he burned up quick.

1

FIGURE 10: A document with a poem in it. The poem is Tim Burton's Stick Boy and Match Girl in Love (BURTON, 1997).


When we need to quote some source code, we can use the verbatim environment:

```
\documentclass{article}
\usepackage[a6paper]{geometry}
\begin{document}
When we need to quote some source code,
we can use the \textsf{verbatim}
environment:
\end{document}
```

1

FIGURE 11: A document with some source code shown in it.

LaTeX provides us with the following commands: `\newcommand`, `\renewcommand`, `\newenvironment` and `\renewenvironment`, which this lesson will not discuss in details. You may find information and examples in OETIKER *et al.* (2018, pp. 104–105). Of course, the *renew* versions are meant to replace a previously defined command or environment.

Anyway, we would like you to analyze an example to understand the basic usage of one of those commands. Let us suppose we have to repetitively write the passage "All work and no play makes Jack a dull boy" in our document.[14]

```
\newcommand\Jack{All work and
no play makes Jack a dull boy}
```

The above definition of a new command tells LaTeX to typeset the text into braces whenever it sees the `\Jack` command in the document. This is the simplest use; we can include TeX code in the definition of a new command so to obtain a more complex result. You can refer, for instance, to STACKEXCHANGE (2016). We can even define commands with mandatory and optional arguments. STACKEXCHANGE (2011) will give you some wonderful explanations and insights about this technique.

14. We guess that Stanley Kubrick and his crew would have loved this facility instead of typewriting that bunch of pages—even localized—for Shining!

8.1.11 *Tables of Contents, Cross References and Indices*

We will see in section 8.2 that when we issue commands like `\chapter` and `\section` we contribute to compile a table of contents. How do we place it in our documents? It is straightforward: just write `\tableofcontents` in the place of your document where you want it to appear. Obviously, you must compile more than once to have the final PDF to be synchronized.

At last, we should talk of two other useful mechanisms that LaTeX offers the authors to write coherent yet complete documents: the cross reference and the indexing.

The cross reference is that technique that allows to refer to whatever element of a document in whatever point of the document. LaTeX has three commands to provide this technique: `\label`, `\ref` and `\pageref`. These commands have a mandatory argument that has to be a unique tag so that labels cannot be mistaken. According to their names, these commands respectively mark a point that will be referred elsewhere in the document; refer to a declared label in the form of a counter (be it a section, a figure number, a footnote. It depends on the point we issue the `\label` command); refer to a declared label in the form of a page number. The labels can be defined and referred in whatever order: they will be correctly referred only after at least the second compilation. A simple example is this:

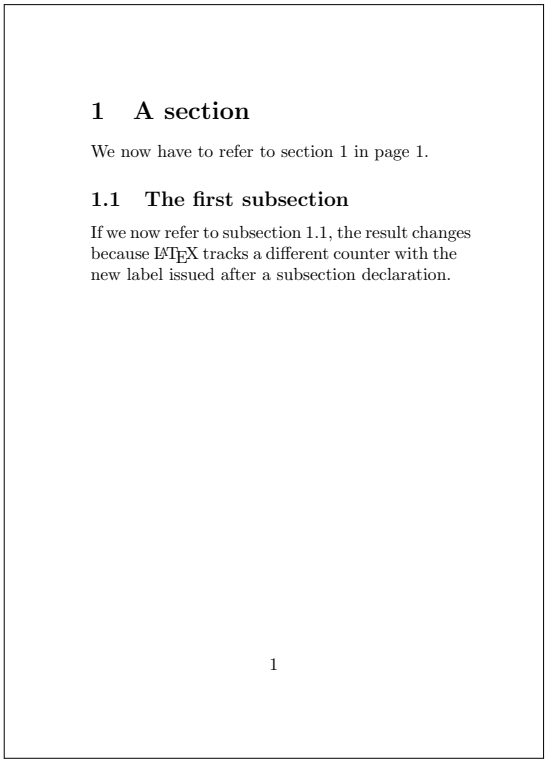


FIGURE 12: This is an example of automatic cross reference performed by LaTeX. After you correctly labeled and referenced the source document, you just have to compile the right number of times to have the right references in the final document. In this way you do not have to worry if you add some text in between: just compile again.

```
\section{A section}
\label{sec:first}
We now have to refer
to section~\ref{sec:first}
in page~\pageref{sec:first}.
\subsection{The first subsection}
\label{sub:first}
If we now refer to
subsection~\ref{sub:first},
the result changes because
\LaTeX\ tracks a different
counter with the new label
issued after a subsection
declaration.
```

and its result is in figure 12. The reference numbers are related to the points labels are issued. Our practice advises us to issue the labels to be referenced either just after the `\chapter`, `\section` and similar commands or in the mandatory argument of the `\footnote` or `\caption` commands. Those labels that will be "pagereferenced" are likely to be put in the exact points they need.

Another sensible task that an author would want to accomplish is the index compilation. Of course LaTeX has a mechanism that allows author to label every term (s)he wants in the index. After labeling, in case of changes in the text, the author just have to recompile at least a couple of times to get everything sorted out.

First of all we have to load a specific package. We strongly recommend Enrico Gregorio's imakeidx, instead of the original makeidx or the slightly more flexible splitidx. We invite you to discover the advantages of using Gregorio's package but as a teaser we can say that you no longer need to compile multiple times to have an updated index.

After we included the package, we may label every term we want with the command `\index`. This command has a mandatory argument: the term you want to appear in the index written exactly in the way you want it in the index. But you should refer to LAMPORT (1987) to discover the power of the LaTeX indexing mechanism and the richness of its syntax.

Now that you labeled your whole document you are ready to include the index in your document: you will issue the command `\printindex` in the point of your document you want the index to appear, compile the document and, depending on the package you are using, you might need to post-process the index file and compile your document again.

8.1.12 Arbitrary Hyphenation

It may be useful, in some special cases where words exceed the right margin, to force an hyphenation. We can do it by typing `\-` in the place we want a specific word to be hyphenated.

8.1.13 An Example of Multilingual Document in XeLaTeX

We close this long section with an example on how easy it is for XeLaTeX to manage languages with alphabets different from Latin. Here you can see some commands already discussed and some that you can read of in ROBERTSON and HOSNY (2017):

```
\usepackage{fontspec}
\setmainfont{IM FELL English}
\usepackage{polyglossia}
\usepackage{xeCJK}
\usepackage{bidi}
\setmainlanguage{italian}
\setotherlanguages{russian,greek,arabic}
\newfontfamily{\russianfont}{FreeSerif}
\newfontfamily{\greekfont}{FreeSerif}
\newfontfamily{\arabicfont}{FreeSerif}
\newcommand\rus[1]{%
  \foreignlanguage{russian}{#1}}
\newcommand\gre[1]{%
  \foreignlanguage{greek}{#1}}
\newcommand\ara[1]{%
  \foreignlanguage{arabic}{#1}}

\begin{document}
Ciao mondo.
```

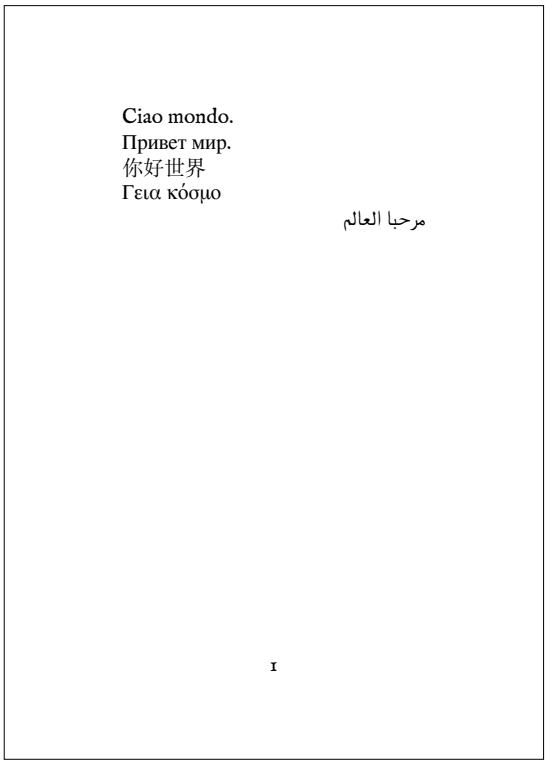


FIGURE 13: The power of fontspec let us get a document like this multilingual Hello, world!.

```
\rus{Привет мир}.

你好世界

\gre{Γεια κόσμο}

\setRTL\ara{ملاعلا ابحرم}
```

Figure 13 shows the related result.

We specified XeLaTeX because LuaLaTeX has different mechanisms and packages to manage Right-to-Left languages. For instance, LuaLaTeX is not compatible with bidi, so the previous example has to be modified to be compiled with LuaLaTeX.

8.1.14 Floating bodies: figures and tables

Depending on the document we are writing, we might have to integrate the text with images and tables. Despite way too many authors want these elements put exactly where they think they should be, LaTeX encourages us to let them float, so that it can keep typesetting the pages according to its quality high standard.

How do we make them float? It suffices to put them in a floating body. LaTeX provides us with two environments that are floating bodies: figure and table. In such environments we will respectively put graphics, drawings (both discussed in a different lesson), images (with the command `\includegraphics` provided by graphicx; this is an extension of the former graphics package) and tables. As for images, graphicx enables us to include JPG, PNG, PDF and EPS images and allow to rotate, clip, trim, dimension and scale images. You may find a complete manual in DAVID P. CARLISLE AND THE LATEX3 PROJECT (2017).

The very basic yet complete usage is

```
\begin{figure}
\centering
\includegraphics{image.pdf}
\caption{Image caption.}
\end{figure}
```

which shows how we add a horizontally centered floating image and caption to our documents putting those elements into a figure container.

Like the previous case, table is just a container that can contain anything, most likely tables. To build a table we need a constructor and the LaTeX basic table constructor is the tabular environment. Instead of explaining, let us see an example, though shortened, related to table 1 and shown in table 2:

```
\begin{table}
\caption{\label{tab:dm-
ex}An example of table construction.}
\begin{tabular}{|c|c|c|}
\hline
\textsc{Command} & \textsc{Example} &
  \textsc{Result} \\
\hline
\verb|\`| & \verb|\`a| & \`a \\
\verb|\'| & \verb|\'a| & \'a \\
\hline
\multicolumn{3}{p{.93\columnwidth}}{It's
possible to have the diacritic mark named
comma below (\c{r}) and its counterpart
inverted comma above (\c{g}) in
\pdfLaTeX\ using the \emph{cedilla}
command (\cmdname{c}), but only with a
selected set of letters. To get the
comma below under s and t you should
keep using the package \pkgname{combelow}
to get such marks.} \\
\hline
\end{tabular}
\end{table}
```

We already know that the pair `\begin{table}-\end{table}` is the floating body that lets the table be placed where LaTeX considers the better place; we also know that the pair `\begin{tabular}-\end{tabular}` is the construction environment. `\begin{tabular}` needs a mandatory argument that specifies how many columns compose the table and their alignment. In our case we see three `c` interleaved by vertical bars; that means 'three centered columns delimited by vertical rules'. Other alignments are `l` (for left), `r` (for right) and `p{`⟨*dimension*⟩`}` (for left aligned in a fixed width column). We can leave out vertical

Table 2: An example of table construction.

Command	Example	Result
\`	\`a	à
\'	\'a	á

It is possible to have the diacritic mark named comma below (ŗ) and its counterpart inverted comma above (ġ) in pdfLaTeX using the *cedilla* command (\c), but only with a selected set of letters. To get the comma below under s and t you should keep using the package combelow to get such marks.

rules, if we do not want them. The command \hline draws horizontal rules.

It is now time to fill the cells. We start writing the content of the first cell (upper left). The content ends when we issue a &. After filling the last cell in a table row we will not issue a & but a line break (\\).

Now, before you ask us why tables 1 and 2 look so different, we should unveil that table 1 has been drawn using tabu, a package that allows a finer control over the table design and look, and booktabs, a package that provides us with finer rulers and better vertical spacing.

The caption has to be issued with the command \caption before we open tabular or after we close it. Please notice the \label in its parameter so that the related reference-counter is correctly referred to a table number.

8.1.15 Colors

LaTeX is not bound to serious black text and white background. Thanks to packages like color or xcolor we can add colors to text, highlight it coloring its background and color a whole page so that the text is on a background different than white. While the color manual is in the graphicx manual (David P. Carlisle and The LaTeX3 Project, 2017), xcolor has a different manual that you can read, as usual, issuing the command `texdoc xcolor` in a terminal.

8.1.16 Again on Special Characters

Now we have a clearer idea of how LaTeX uses some characters: \ starts a command, so it cannot be used *per se* to represent a backslash; ~ is a special space, so it cannot be used to represent a tilde character; & is a tabbing character in tables and cannot represent the ampersand character; % starts a comment.

Tilde can be obtained as \~ (~, maybe not exactly in the position we expected it), as $\sim$ (∼, quite big, huh?) and as \textasciitilde via textcomp (~, like in the first case). Ampersand is \& and percent is \%.

What about the backslash, since \\ breaks a line? Once again textcomp helps: \textbackslash (\).

Table 3: LaTeX commands to give documents a structure.

Command	book	report	article
\part	✓	✓	✓
\chapter	✓	✓	
\section	✓	✓	✓
\subsection	✓	✓	✓
\subsubsection	✓	✓	✓
\paragraph	✓	✓	✓
\subparagraph	✓	✓	✓

8.2 Main Body Analysis: Document Structure

The structure of a document is directly related to the type of the document. As we have mentioned in section 7.1, the way a book is organized is slightly different than the way an article is organized, and both of them totally differ from a letter. Roughly speaking, a document structure is directly related to the text organization.

When we organize a document, being it a book, a report or an article, we subdivide it into several main topics (chapters for books and reports, sections for articles). They could be grouped into parts, but not necessarily. This leads to a special treatment of counters that we are going to talk about in a few lines. A main part will probably be subdivided into smaller blocks of text, each one treating a specific subtopic: sections for books and reports, subsections for articles. Every smaller block can be subdivided into smaller subblocks. When we organize our documents according to these blocks and subblocks—levels—, we are giving our documents a structure. Table 3 lists the commands provided by LaTeX to structure documents and indicates which classes use them or not.

All of these commands take a mandatory argument (the title) and an optional argument (the title as it will be included in the table of contents. If no optional argument has been indicated, the main title will be included in the table of contents). The commands add a number before every title. Unless we change the way the numbers are stored, they have the following forms: part number, chapter number, [chapter.]section number (the brackets mean that there is no chapter numbers in an article), [chapter.]section.subsection number and so on.

As you can figure out, this "strategy" helps authors not to make errors with the structure. Suppose we are writing an article. After the \section, instead of declaring a \subsection, we declare a \subsubsection as in the following example:

```
\begin{document}
\section{Section title}
```

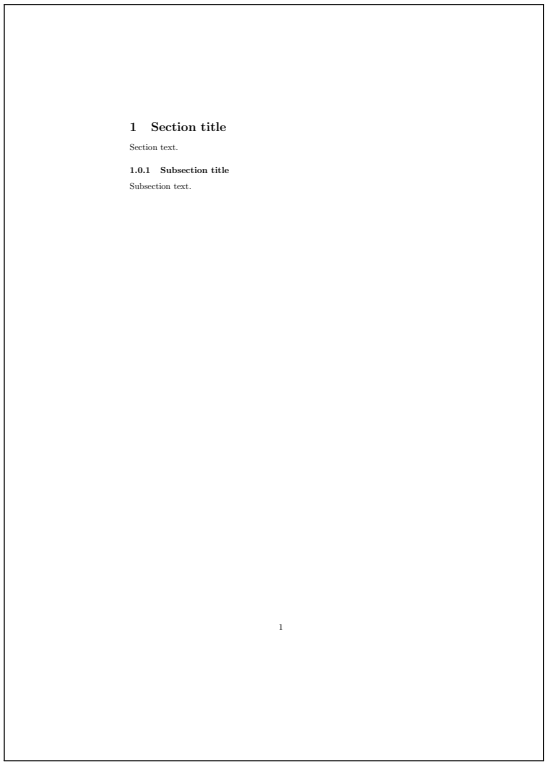


FIGURE 14: The number 1.0.1 indicates that we skipped a level in the structure.

```
Section text.

\subsubsection{Subsection title}
Subsection text.
\end{document}
```

As we can see in figure 14, what we expected to be numbered 1.1 is instead numbered 1.0.1. This means that we skipped a level: instead of using a `\subsection` command, we used a `\subsubsection`.

All of the commands we mentioned in this section have a corresponding starred version (i.e., `\section*`). These versions do not print the number before the corresponding title and do not let the titles appear in the table of contents.

It is a good practice, when we organize the structure, to let (let us say) a section have two or more subsections, if subsections are necessary.

8.3 Splitting Big Documents

When writing big documents such as books, we could find handy to split the LaTeX source into different files. How do we put them all together?

The most common case is a master file that includes slave documents, for instance one file per chapter in the document body.

LaTeX offers two ways to include such files: `\include{`⟨*filename*⟩`}` (without the `.tex` extension) starts a new page, separately processes the file and includes it in the final camera ready; `\input{`⟨*filename*⟩`}` (without the `.tex` extension) that simply processes the file as being part of a single, big file.

You will find further information on the fine control of such an operation in Oetiker *et al.* (2018) or, better, in Kopka and Daly (2004).

8.4 Help, I Need a Symbol

If you spent many hours writing LaTeX documents, you may have experienced that you cannot recall the name of a specific symbol you really need now. Reading Pakin (2017) could be helpful, but you are not sure you can really locate that symbol in a manual longer than 300 pages. How can you do?

Thanks to Philipp Kühl and Daniel Kirsch, we may rely on a website (Kühl and Kirsch, 2019) that allows us to sketch the symbol we are looking for and answers with a list of those symbols that look like our sketch along with the related commands.

Bonus Section: Guess What!

Many of you, experienced with LaTeX, know the usual aspect of LaTeX documents. You even know the power and the lacks of TeX. So we decided to deliver you a game: figures 15–31 show some books, journals, magazines and report pages along with other documents. You are invited to guess which of them have been typeset with LaTeX and which of them have not been. The solution is at the end of the lesson.

Part III: Writing LaTeX Documents

9 (Not Necessarily) Dedicated Editors

We already know that users do not need a special editor to write a LaTeX document. We will not stress enough the fact that users tend to think in terms of tools; the risk is that a user who normally edits LaTeX documents with, say, Kile, gets lost without it and does not consider to use a different IDE or text editor or even does not consider the fact that a different editor can be used (can we call it the Word syndrome?).[15] In spite of that, LaTeX users can rely on several more or less dedicated editors.

The first one, one of the most famous tools in the Unix world, is Emacs. It is not especially written to integrate with LaTeX, but with AUCTeX it can be used as a powerful IDE (integrated development

15. [NdGP] When I used to write programs with Borland Turbo Pascal and Turbo C, I could not understand that I was allowed to edit source codes with whatever editor, so I always opened the IDE not only to get an executable program but even because I thought I could not edit the source code in a different way, just like any proprietary environment. Then I switched to Unix...

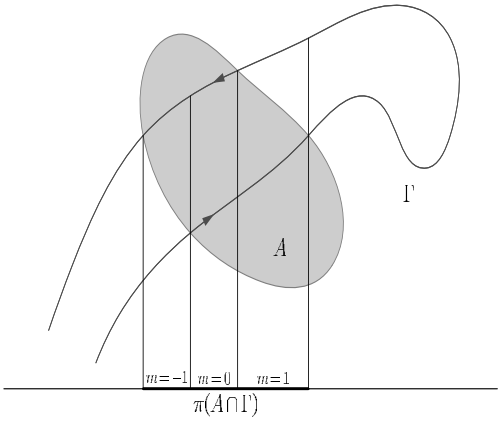


Figura 3.3: Molteplicità algebrica

$\varphi(A,\pi)$ è esprimibile mediante un integrale superficiale nel modo seguente:

$$\varphi(A,\pi)=\int_{A\cap\Gamma}\langle\mathbf{n},\mathbf{n}_\pi\rangle\,d\sigma_2,$$

ove $\mathbf{n}$ è il versore normale di Γ. Quindi ciascuna di queste funzioni fornisce, localmente, una stima per difetto dell'area.

Nel caso in cui $\Gamma=\partial E$, scegliendo $\pi=\pi_{yz}$ otteniamo

$$\int_E\frac{\partial g}{\partial x}\,dx\,dy\,dx=\int_\Gamma g\,d\varphi(\cdot,\pi_{yz})\qquad\forall\,g\in C_c^1(\mathbf{R}^3).$$

Nella terminologia moderna $\varphi(\cdot,\pi_{yz})$ è quindi la derivata *nel senso delle distribuzioni* della funzione caratteristica χ_E lungo la direzione x (e analogamente $\varphi(\cdot,\pi_{xy})$ e $\varphi(\cdot,\pi_{zx})$).

FIGURE 15: Mathematical formulae and diagrams.

BIBLIOTECA
LEONARDIANA
STUDI E DOCUMENTI
5

SCIENZE E RAPPRESENTAZIONI

SAGGI IN ONORE DI PIERRE SOUFFRIN

Atti del convegno internazionale
Vinci, Biblioteca Leonardiana, 26-29 settembre 2012

a cura di
Pierre Caye, Romano Nanni e Pier Daniele Napolitani

LEO S. OLSCHKI EDITORE
MMXV

Figure 16: Frontispiece of a proceedings volume, published by Olschki.

beam in terms of thickness and matter mean both equal and similar in Arabic. There is a clear preference in Arabic mathematical texts for using the first for equal and the second for similar. Thus, Knorr translated them in this manner (KNORR 1982, p. 139). In the given context it is clear though that similarity is not meant literally, but in the sense of having the same property. This ambiguity reflects the use of ἴσος and ὁμοίος for respective concepts in Greek.

5.2. *Investigation 2*

Liber de Canonio, Proposition II

Si fuerit proportio ponderis in termino minoris portionis suspensi, ad superhabundantiam ponderis maioris portionis ad minorem, sicut proportio longitudinis totius canonii ad duplam longitudinis minoris portionis, erit canonium parallelum epipedo orizontis (MOODY & CLAGETT 1952, p. 66).

If the proportion of the weight suspended at the end of the smaller portion to the surplus of the weight of the greater portion to the smaller will be like the proportion of the length of the entire beam to the double of the length of the smaller portion, the beam will be parallel to the surface of the horizon (Cf. MOODY & CLAGETT 1952, p. 67).

MS Beirut, *ziyāda*, Proposition 4

اذا كان عمود متساوي الغلظ متشابه الجوهر وقسم بقسمين مختلفين وعلق بنقطة طرف القسم الاقصر ثقل وجعلت نسبة الثقل الى ثقل فضل القسم الاطول على ثقل القسم الاقصر كنسبة نصف طول العمود كله الى طول القسم الاقصر فان العمود يعتدل على موازاة الافق.

(KNORR 1982, p. 154).

If there is a beam, (which is) equal in itself in thickness, equal in itself in substance and partitioned in two different parts and (if) a weight is suspended at the end of the shorter part and the ratio of the weight to the weight of the surplus of the longer part over the weight of the shorter part is made like the ratio of half of the length of all of the beam to the length of the shorter part, then the beam equilibrates itself in parallelness to the horizon.

Again, the content of both theorems is the same and the two enunciations are similar, but not identical. Their difference is greater than in the previous case, because the *Liber de canonio* does not repeat the description of the properties of the beam and the suspended weight and thus has to integrate the latter into the description of the proportion. It differs from the *ziyāda* also in regard to the placement of the term *weight* in the description of the second term of the proportion. The *Liber de canonio* uses the term only once between *superhabundatiam* and *maioris*. The *ziyāda* uses it twice, once before the surplus and once before the shorter part. While the formulation of the *Liber de canonio* is imprecise, but comprehensible, the formulation of the *ziyāda* is comprehensible, but false. It is most likely the result of a scribal error as

FIGURE 17: Multilingual parallel texts, from the same volume.

Paola Manni

SULLA TERMINOLOGIA DELLE MACCHINE IN LEONARDO: TRADIZIONE, INNOVAZIONE E SVILUPPI FUTURI*

Qui si dimostra la natura della vite e di sua lieva,
e chome ella debbe più tosto ess(er)e adop(er)ata <in is>
in tirare che in ispingiere. E chom'ella fa più for-
ça a essere senplice che doppia, e sottile che grossa,
essendo mossa da pari lungeça di lieva e pari força.
E chosì si farà un pocho di discorso in qua(n)ti modi si
pò adop(er)are, e di qua(n)te sorte si pò fare viti sança
fine. E qua(n)ti moti son fatti sança vite, che fa(n)-
no p(r)opio ofitio di vite. E in che modo la vite
sança fine s'achonpagni cholle rote dentate, e
chome molte viti si debono insieme adop(er)are.
E ssi dirà della natura delle sue madri, e sse so(n)
più utili cho· molti denti o nno. E si dirà delle
viti retrose e delle viti che p(er) un medesimo ti-
rare spingano e ttirano il peso, e di viti che
p(er) una sola volta che se le dia, farà fugire la sua
madre molte delle sue volte circulari. E così
moltissimi sua effetti, e varie fatiche, e fforteçe,
e tardità, e p(r)esteçe. E ssi prov(er)rà ragio(n)e[1] <di ut>
di tutti loro ofiti e nature, e materie, e llieve,
e utilità. E ssi dirà in che modo si debbono fare,
e del modo del metterle in op(er)a;
e di chi è stato inganato p(er) no(n) cognosscer lor natura.
E ttali strume(n)ti si figurera(n)no in gra(n) parte sança
le loro armadure, o altra cosa che avessi a inpe-

* Le trascrizioni dai codici leonardiani sono fatte seguendo le norme stabilite da Arrigo Castellani per l'edizione dei testi medievali, già utilizzate in Manni 2008 e in Manni & Biffi 2011. Alle pagine introduttive di quest'ultimo (pp. xxxi-xxxii) si rimanda per una loro esposizione dettagliata e ulteriori riferimenti bibliografici. Nel caso di citazioni brevi inserite nel corpo del testo, si eliminano le parentesi tonde che segnalano lo scioglimento delle abbreviazioni. Con la sigla Madrid I si indica il primo codice di Madrid (Biblioteca Nacional de España, cod. 8937).

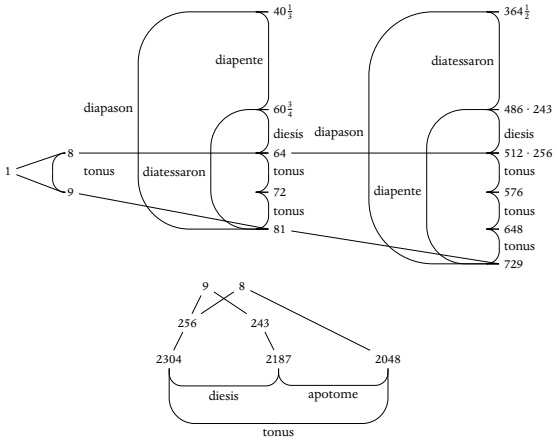
[1] La *e* non chiara, corretta su altra lettera.

Figure 18: Automatic line numbering, from the same volume.

17^{am}, 18^{am}, 19^{am}, 20^{am}. Itaque deinceps fieri potest in infinitum.

63 Hinc patet origo numeri harum vocum hexachordum constituentium, scilicet *ut re mi f a sol la*. Octo autem literae Γ *a b c d e f g* statutae sunt, ut earum unaquaeque octavo quoque loco repetita diapason consonantiam in proportione dupla semper indicet; quod numeri in singulis chordis icosichordi dispositi, sicut omnes alias consonantias et spacia, ostendunt. **64** *g* littera, quia sonora, dat initium hexachordo ♮ quadri et duri. *c*, quoniam media inter aspiratam et sonoram, dat initium hexachordo naturae diatonici generis. *f*, quoniam sapit ipsa aspiratam et mollem, dat initium hexachordo ♭ mollis et chromatici generis.

65 Continuatio autem tonorum in sesquioctava proportione et constitutio sesquialterae ac sesquitertiae proportionum, hoc est diatessaron ac diapente componentium diapason, ac dieseos spacium relinquentium, sic patet in numeris:



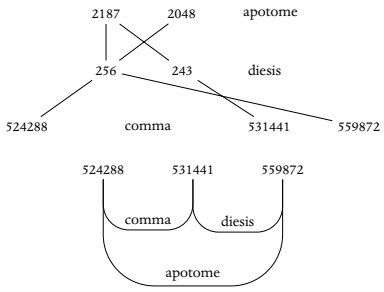
66 Ex quibus constat quod diesis proportio est in his numeris: 256 — 243. Videnda est A:32v
nunc proportio semitonii maioris sive apotomes. Sic proportio 9 — 8 facit tonum; ducatur 9 in 256 et mox in 243 et fiunt duo numeri 2304, 2187; quorum proportio est sicut 256 — 2(4)3, scilicet diesis. Item ducatur 256 in 8 et fiat 2048. Eritque sicut 9 — 8, sic 2304 — 2048. Quare proportio 2304 — 2048 faciet tonum cumque proportio 2304 — 2187 faciat diesim, sive semitonium minus, supererit proportio 2187 — 2048, semitonii maioris scilicet apotomes.

67 Di erentia vero diesis et apotomes dicitur comma, quod elicitur per subtractionem unius proportionis ab alia, ut infra patet. A:33r

3 *ante* Γ *del.* .a.b.c. *A* **6** *g* littera ∼ chromatici generis *in laevo inf. marg. A* **8** ipsa: ip *A*

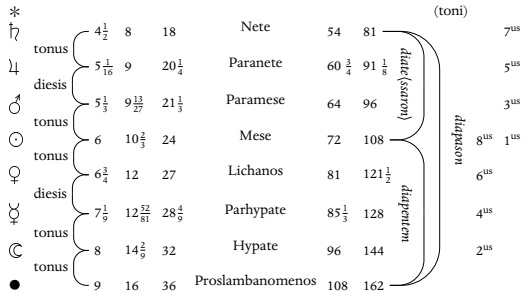
[r- v]

FIGURE 19: Diagrams from the critical edition of Francesco Maurolico's *Musica*.



[68] Constabit etiam quod diesis maior est, quam tria commata minor autem quam quatuor.
[69] Apotome autem maior, quam quatuor commata, minor quam quinque.
[70] Unde et tonus excedet octo commata et minor quam novem commata nascitur, quae omnia ex longo et multarum figurarum calculo constare possunt lege Boetium et Fabrum in musicis elementis.

O



Die ☾ 30 decembris 1566.

4 *ante* possunt *del.* pãt *A*

[70] lege Boetium: ◇ Fabrum in musicis elementis.:

FIGURE 20: More diagrams from the critical edition of Francesco Maurolico's *Musica*.

affermazione dalla definizione di limite e "da quelle successive": a tali definizioni Cantor accenna soltanto, ma si possono svolgere in modo naturale, come nelle esposizioni moderne.

L'uguaglianza, la relazione d'ordine e le operazioni sono definite per punti (*pointwise*). L'uguaglianza è definita da Cantor, come abbiamo visto, e comporta che se $b = \lim a_n$ e $b' = \lim a'_n$ allora $b = b'$ se e solo se

$$\forall \varepsilon > 0 \exists n_0 \forall n > n_0 (\mid a_n - a'_n \mid < \varepsilon).$$

Se $b = \lim a_n$ e $b' = \lim a'_n$ allora $b + b' = \lim(a_n + a'_n)$, dopo aver dimostrato che $\{a_n + a'_n\}$ è di Cauchy; come caso particolare, se $b = \lim a_n$ e $r \in \mathbb{Q}$ allora $b + r = \lim(a_n + r)$, dopo aver dimostrato che $\{a_n + r\}$ è di Cauchy.

Analogamente $b \leq b'$ se e solo se $\exists n_0 \forall n > n_0 (a_n \leq a'_n)$; in particolare $b \leq r$ se e solo se da un certo punto in poi $a_n \leq r$.

La relazione $<$ deve essere definita come "$\leq$ e $\neq$", che equivale a dire, se $b = \lim a_n$ e $b' = \lim a'_n$:

$$b < b' \text{ se e solo se } \exists \varepsilon > 0 \; \exists n_0 \forall n > n_0 (a'_n - a_n > \varepsilon).$$

Se $b = \lim a_n$ e $r \in \mathbb{Q}$, $b < r$ se e solo se esiste un $\varepsilon > 0$ tale che da un certo punto in poi $r - a_n > \varepsilon$.

Si dimostra la tricotomia, vale a dire che per b e b' o razionali o simboli di irrazionali associati a successioni di Cauchy

$$b = b' \text{ o } b < b' \text{ o } b' < b.$$

Ora per ogni razionale r, scriviamo (r) per indicare la successione costante $\{r, r, \ldots\}$. Sia $b = \lim a_n$ e per ogni n fissato confrontiamo b con la successione (a_n). Si vuole dimostrare che per ogni $\varepsilon > 0$ (ci si può restringere a ε razionali), almeno da un certo punto in poi $\mid b - (a_n) \mid < \varepsilon$, che coinvolge solo relazioni e operazioni algebriche già definite per i nuovi numeri. $\mid b - (a_n) \mid < \varepsilon$ significa che

$$\exists m_0 \forall m > m_0 (\mid a_m - (a_n)_m \mid < (\varepsilon)_m).$$

Siccome la successione $\{a_n\}$ è di Cauchy, per ogni $\varepsilon > 0$ razionale

$$\exists m_0 \forall n > m_0 \forall m > m_0 (\mid a_m - a_n \mid < \varepsilon),$$

quindi

$$\exists m_0 \forall n > m_0 \forall m > m_0 (\mid a_m - (a_n)_m \mid < (\varepsilon)_m),$$

che è quello che si voleva dimostrare.

Si noti che viceversa, se $b = \lim a_n$ e la successione $\{a'_n\}$ è tale che $\forall \varepsilon > 0 \exists n_0 \forall n > n_0 (\mid b - a'_n \mid < \varepsilon)$ allora, prendendo $\varepsilon/2$ qui e in $\lim a_n$ si ha per n sufficientemente grande $\mid a_n - a'_n \mid < \varepsilon$, da cui $\lim a'_n = b$.

FIGURE 21: A page from a book on the development of mathematical logic.

20 A signo enim lucidi cuiuspiam *A* in planum *BC* cadant radii *AB*, *AD*, *AE*, *AF* et *AC*, intelliganturque iidem radii rem *GH*, signo *A* propiorem, illuminare in signis *G*, *H* , *K*, *L*, *M*, sitque *AE* radius ipsis *BC* et *GH* planis perpendicularis; ipsi vero *AD* et *AF* item et *AB* et *AC*, aequales. Aio quod *BE* et *EC* plana aequaliter, ipsum vero *GH* magis illuminabitur. Patet enim radios *AB*, *AD*, *AE*, *AF* et *AC* in planis *BE* et *EC* esse aequaliter densos, in plano vero *GH* densiores.⟦*Sunt enim spatia DE, BD spatiis FE, CF aequalia, quibus quidem minora sunt spatia KL, GK, ML, HM.* ⟧ Igitur per secundum suppositum, ipsa *BE* et *EC* plana aequaliter, ipsum vero *GH* magis illuminabitur.

21 Quod si intelligatur | signum *A* spatio *BC* propius fieri inter ipsas *BA*, *AC* lineas, crescet iam *BAC* angulus. Itaque *BC* spatium plures suscipiet radios. Quare ex quinto supposito magis illuminabitur.

C

22 Hinc et illud sequitur, ut sol aeque a se remota aequaliter, propiora vero magis calefaciat.

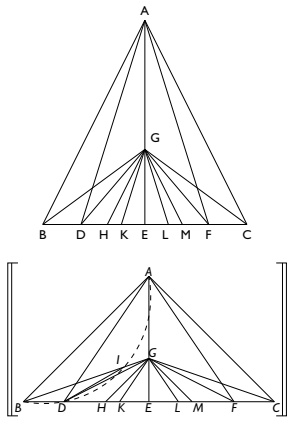
‖ T 4

23 Potest signum plano tantum propinquare, ut planum ipsum fortius, verum particularius illuminet.

24 Signum *A* planum *BC* illuminet radiis *AB*, *AD*, *AE*, *AF* et *AC*, e quibus *AE* perpendicularis. Aio quod possibile est signum *A* tantum propius fieri plano *BC*, ut magis per minorem ipsius plani partem illustret. Fiat enim propinquius signum *A* ipsi *BC* plano in signo *G* lineae *AE*, ita ut ductis radiis *GB*, *GD*, *GE*, *GF* et *GC*, angulus *BGD* minor fiat angulo *BAD*, hoc enim possibile est.

25 ⟦*Et demonstratur: nam si ex scholio propositionis 5ae libri 4i per tria puncta B, D, A describatur arcus circuli BDA et extra arcum in recta AE accipiatur punctum infra G, a quo ducantur rectae GB, GD et reliquae, ut modo dictum est; item ex puncto I, ubi recta BG secat arcum DA DA, ducatur recta DI. Erit angulus BID aequalis angulo BAD per 21*am *tertii, et maior angulo BGD per 16*am *primi.*⟧

26 Ducantur etiam *GH* ipsi *AB*, et *GK* ipsi *AD*, paralleli; item *GL* ipsi *AF*, et *GM* ipsi *AC*, paralleli radii. Ergo sub angulo *BAC* aequales sunt numero radii radiis sub angulo *HGM* comprehensis. Sed hi densiores,⟦*quia minus spatium occupant.* ⟧ Igitur per secundum suppositum erit planum *HM* illustratius plano *BC*. Radii vero sub angulo



15 signum *S* spacium *C* **33** radiis *S C*2 mediis *C* **36–43** Et demonstratur ∼ primi. *S*1

38 *BDA correximus BDGA add.* *S*1 ◇ *AE correximus GE* *S*1 **41** *DA correximus DG* *S*1

21 per secundum suppositum: *Maur. Phot. Supp. 2* Densiores radios intensius, aeque vero densos aequaliter illuminare. ◇ ex quinto supposito: *Maur. Phot. Supp. 5* Plures radios intensius, aequales vero aequaliter illuminare. **25** ex scholio propositionis 5ae libri 4i: *Clav. Elem.*, I, pp. 471 sg.

◇ per 21am tertii: *Clav. Elem.*, I, pp. 410 sg. In circulo qui in eodem segmento sunt anguli sunt inter se aequales. ◇ per 16am primi: *Clav. Elem.*, I, p. 188 sg. Cuiuscumque trianguli uno latere producto, externus angulus utrolibet interno et opposito maior est. **26** per secundum suppositum: *Maur. Phot. Supp. 2* Densiores radios intensius, aeque vero densos aequaliter illuminare.

[C v- r · S -]

FIGURE 22: Geometric diagrams from the critical edition of Francesco Maurolico's *Optica*.

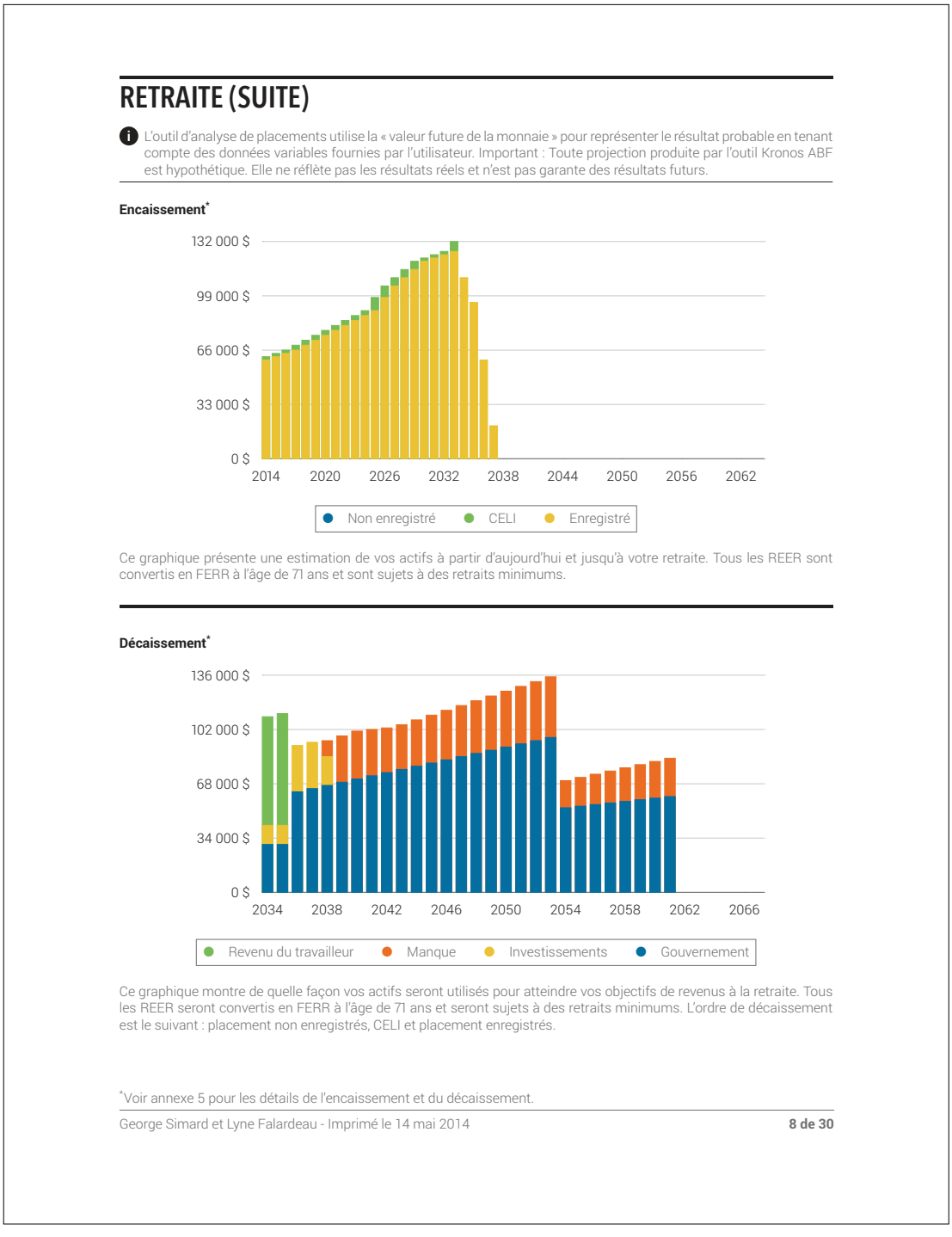
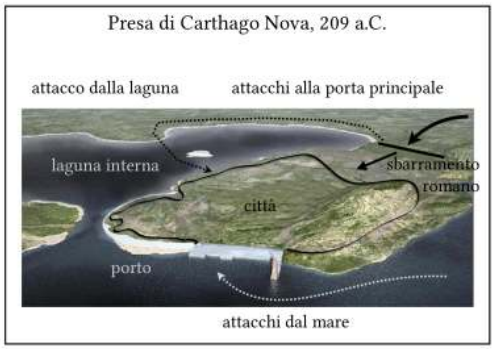


FIGURE 23: Graphics from a financial report.



Disciplina e mistica di un capo

Scipione avviò subito un programma di preparazione delle truppe per rendere più elastica la formazione classica erede della falange, che basava la sua efficacia nella forza d'urto e richiedeva di ampie pianure per rendere al meglio. Il giovane capitano si preparava a emulare le manovre avvolgenti dispiegate dalle truppe di Annibale trovandosi ad addestrare un esercito abituato a lottare in un altro modo. Oltre a migliorare la tecnica di difesa contro arcieri e frombolieri, Publio divise la legione in piccole unità (manipoli) in grado di incalzare il nemico con attacchi in profondità ma anche in grado di ritornare agilmente sui propri passi. A questo si aggiungeva l'adozione della spada iberica, più corta e adatta al combattimento uno contro uno, nel quale i guerrieri iberici erano noti per il loro *furor*. Probabilmente sviluppò anche la formazione della coorte, unità intermedia fra la legione e il manipolo, in grado di replicare la forza d'urto dello schieramento completo e capace di cambiare rapidamente la formazione.

L'allenamento seguiva un calendario preciso che combinava la corsa in tenuta di battaglia, pulizia e manutenzione delle armi, riposo e pratica con le armi. Scipione ordinò altresì la ripresa delle attività normali nella città e il riatta-

Figure 24: A page from an EDUSC series.

FIGURE 25: The dust cover jacket of one of the authors' book.

but not always possible on lesser terminals like that built into a PDA, phone or even some basic telnet interfaces.

Conclusion

Today we are all familiar with using a GUI interface for the majority of our work, from web browsers to office applications and email. However, there are times when text based is what you need. In my case, the only service I could get to work at one point last week was a dial-up connection through a bulletin board to my hosted server, using a mobile phone while in an airport in Europe; all for the benefit of discussing a project with a client in the US.

A terminal based solution wouldn't be my first choice, but a quick test of a few applications showed there is a lot of choice out there. Fortunately, a terminal based application does not mean limited or restricted. In fact, there's very little I found I couldn't do with these text-based packages, especially for basic and straightforward discussions.

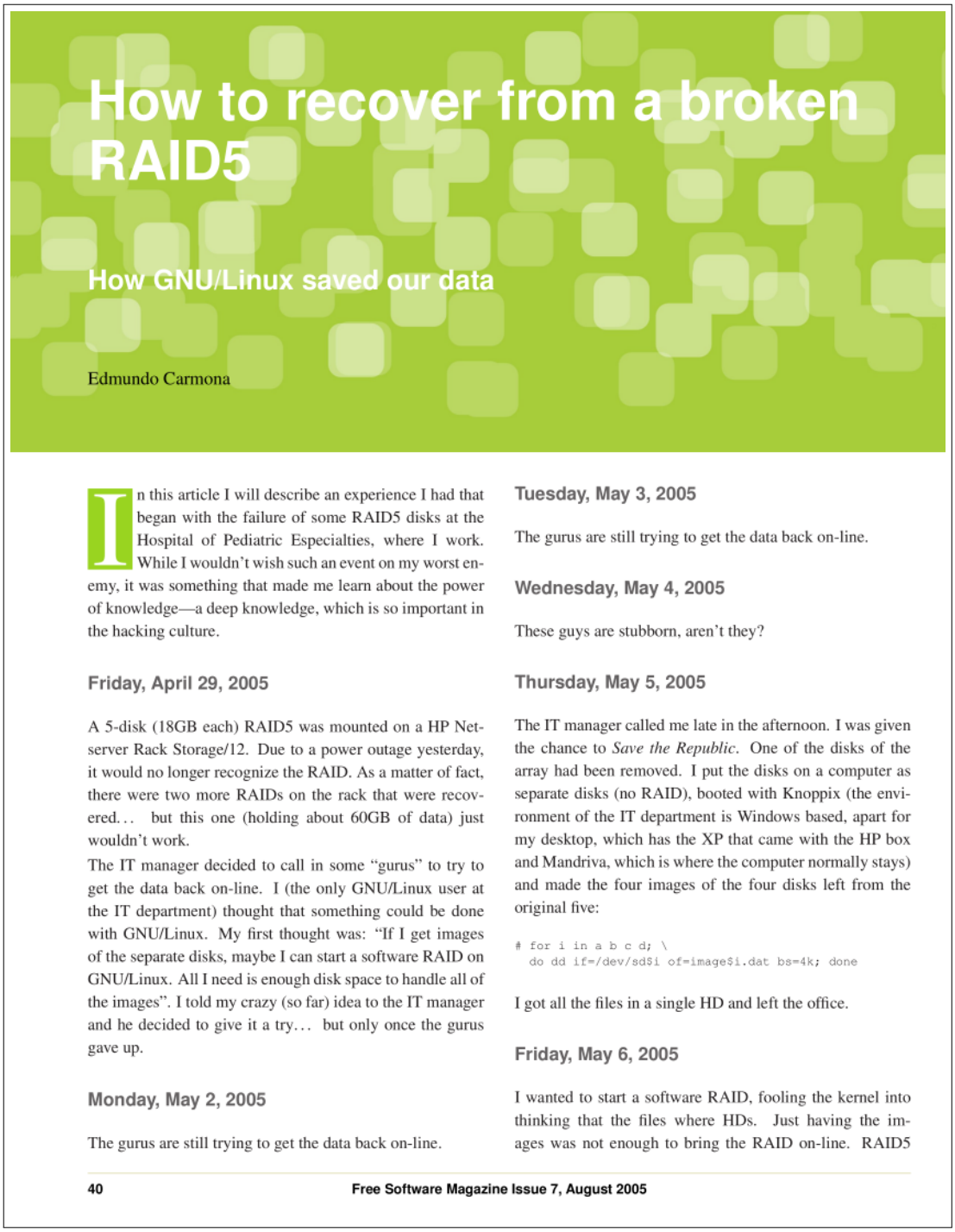
As to choices, in an ideal world with a nice large monitor I'd choose CenterICQ, only because it would simplify the connectivity to other applications. However, for a good all-purpose IRC only client that could download and use pretty much everywhere, I'd pick Rhapsody.

Copyright information

About the author

Martin "MC" Brown is a freelance writer and consultant, he works with Microsoft as an SME, is a featured blogger for ComputerWorld, a founding member of AnswerSquad.com, Technical Director of Foodware.net and, and has written books on topics as diverse as Microsoft Certification, iMacs, and free software programming.

FIGURE 26: One page from Free Software Magazine n. 7 (camera ready for Lulu.com).



How to recover from a broken RAID5

How GNU/Linux saved our data

Edmundo Carmona

In this article I will describe an experience I had that began with the failure of some RAID5 disks at the Hospital of Pediatric Especialties, where I work. While I wouldn't wish such an event on my worst enemy, it was something that made me learn about the power of knowledge—a deep knowledge, which is so important in the hacking culture.

Friday, April 29, 2005

A 5-disk (18GB each) RAID5 was mounted on a HP Netserver Rack Storage/12. Due to a power outage yesterday, it would no longer recognize the RAID. As a matter of fact, there were two more RAIDs on the rack that were recovered... but this one (holding about 60GB of data) just wouldn't work.

The IT manager decided to call in some "gurus" to try to get the data back on-line. I (the only GNU/Linux user at the IT department) thought that something could be done with GNU/Linux. My first thought was: "If I get images of the separate disks, maybe I can start a software RAID on GNU/Linux. All I need is enough disk space to handle all of the images". I told my crazy (so far) idea to the IT manager and he decided to give it a try... but only once the gurus gave up.

Monday, May 2, 2005

The gurus are still trying to get the data back on-line.

Tuesday, May 3, 2005

The gurus are still trying to get the data back on-line.

Wednesday, May 4, 2005

These guys are stubborn, aren't they?

Thursday, May 5, 2005

The IT manager called me late in the afternoon. I was given the chance to *Save the Republic*. One of the disks of the array had been removed. I put the disks on a computer as separate disks (no RAID), booted with Knoppix (the environment of the IT department is Windows based, apart for my desktop, which has the XP that came with the HP box and Mandriva, which is where the computer normally stays) and made the four images of the four disks left from the original five:

```
# for i in a b c d; \
  do dd if=/dev/sd$i of=image$i.dat bs=4k; done
```

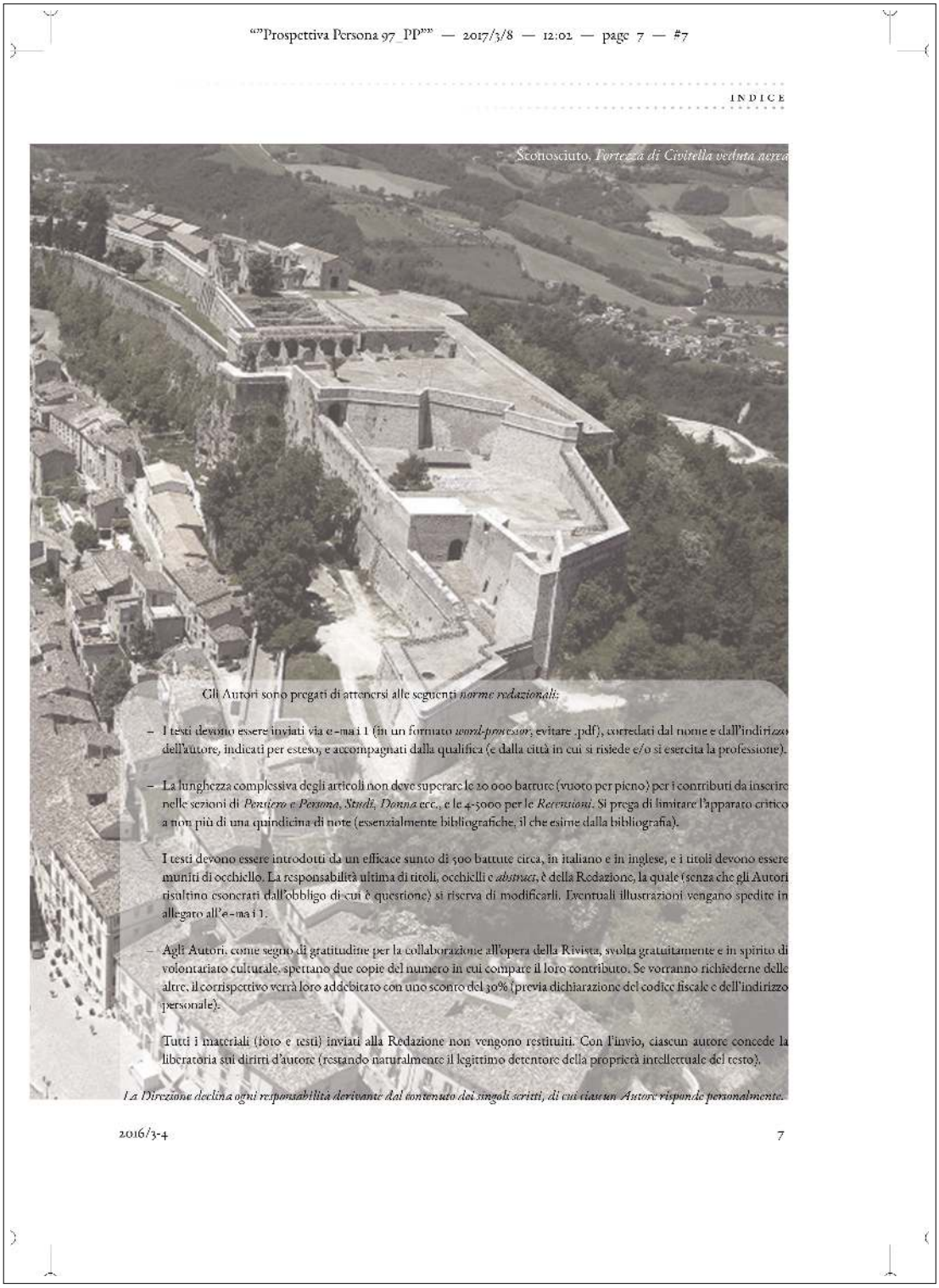
I got all the files in a single HD and left the office.

Friday, May 6, 2005

I wanted to start a software RAID, fooling the kernel into thinking that the files where HDs. Just having the images was not enough to bring the RAID on-line. RAID5

40 Free Software Magazine Issue 7, August 2005

FIGURE 27: Another page from Free Software Magazine n. 7.



""Prospettiva Persona 97_PP"" — 2017/3/8 — 12:02 — page 7 — #7

INDICE

Sconosciuto, *Fortezza di Civitella veduta aerea*

Gli Autori sono pregati di attenersi alle seguenti *norme redazionali*:

– I testi devono essere inviati via e-mail (in un formato *word-processor*, evitare .pdf), corredati dal nome e dall'indirizzo dell'autore, indicati per esteso, e accompagnati dalla qualifica (e dalla città in cui si risiede e/o si esercita la professione).

– La lunghezza complessiva degli articoli non deve superare le 20 000 battute (vuoto per pieno) per i contributi da inserire nelle sezioni di *Pensiero e Persona*, *Studi*, *Donna* ecc., e le 4-5000 per le *Recensioni*. Si prega di limitare l'apparato critico a non più di una quindicina di note (essenzialmente bibliografiche, il che esime dalla bibliografia).

I testi devono essere introdotti da un efficace sunto di 500 battute circa, in italiano e in inglese, e i titoli devono essere muniti di occhiello. La responsabilità ultima di titoli, occhielli e *abstract*, è della Redazione, la quale (senza che gli Autori risultino esonerati dall'obbligo di cui è questione) si riserva di modificarli. Eventuali illustrazioni vengano spedite in allegato all'e-mail.

– Agli Autori, come segno di gratitudine per la collaborazione all'opera della Rivista, svolta gratuitamente e in spirito di volontariato culturale, spettano due copie del numero in cui compare il loro contributo. Se vorranno richiederne delle altre, il corrispettivo verrà loro addebitato con uno sconto del 30% (previa dichiarazione del codice fiscale e dell'indirizzo personale).

Tutti i materiali (foto e testi) inviati alla Redazione non vengono restituiti. Con l'invio, ciascun autore concede la liberatoria sui diritti d'autore (restando naturalmente il legittimo detentore della proprietà intellettuale del testo).

La Direzione declina ogni responsabilità derivante dal contenuto dei singoli scritti, di cui ciascun Autore risponde personalmente.

2016/3-4 7

FIGURE 28: Prospettiva Persona editorial rules.

cuno che in un certo modo non si confonde mai totalmente con la colpa e con il momento della colpa passata, e neppure con il passato in generale»[22]? Il passato non passa, come visto. Il tempo non può cancellare il fatto di aver fatto. E tuttavia il perdono è capace di sciogliere il legame tra l'agente e il fatto da lui compiuto, non nel senso di indebolire la responsabilità di questi, che resta, quanto piuttosto nel senso di non confinare l'agente a quel solo agito. La persona dell'agente, pur restando definita dal male commesso, grazie al *presente* del perdono, viene sciolta dal nodo che la lega indissolubilmente a ciò che ha fatto e a ciò che è stata, al *passato*, per aprire un *futuro* che non sia la mera ripetizione del già stato, ma la ripresa di sé in libertà, il recupero della propria dignità calpestata, l'essere riportata all'altezza da cui è decaduta. Afferma la Arendt:

> senza essere perdonati, liberati dalle conseguenze di ciò che abbiamo fatto, la nostra capacità di agire sarebbe per così dire confinata a un singolo gesto da cui non potremmo mai riprenderci; rimarremmo per sempre vittime delle sue conseguenze, come l'apprendista stregone che non aveva la formula magica per rompere l'incantesimo[23].

Ma il tempo interviene anche nella maturazione verso la concessione del perdono. Perdonare non è qualcosa di magico, ma un processo che richiede tempo e fatica. Questo processo può avere inizio soltanto quando, superata una *prima fase* di smarrimento e sofferenza personale, ingiusta e profonda, la persona, attraversata una *seconda fase* di rancore, risentimento, se non odio, giunge ad una *terza fase*, in cui recupera un certo controllo sui propri vissuti e sulla propria sofferenza: non li nega, ma neppure se ne lascia dominare[24]. Si riprende dal colpo subito e si fa quindi capace di andare oltre l'accaduto, perché guarda all'offensore come capace di essere di più del male commesso. Sostanzialmente il perdono presente, andando oltre l'accaduto passato, guarda al futuro nella convinzione maturata che l'altro, l'offensore, non è circoscritto al male compiuto. Il perdono, nel gesto di misericordia incondizionata che è, esprime una fiducia nell'altro che apre alla speranza, che può affermare: «tu vali molto di più delle tue azioni»[25], tu *puoi* essere all'altezza di te stesso e di quello che vali.

Perdono e riconciliazione

Queste considerazioni, tuttavia, non ci devono spingere a confondere perdono con riconciliazione. Il perdono, lo abbiamo visto, resta incondizionato e, se accettato, *può* riportare la persona dell'offensore all'altezza di se stesso e della sua dignità. Ma potrebbe anche accadere che tutto il gioco del senso di colpa più sopra preso in rapido esame potrebbe non

Immagine 14: Gianlorenzo Bernini, *Ermafrodito dormiente*, 1620, Parigi, Museo del Louvre

[22] *Ivi*, p. 30. [23] Hannah Arendt, *Vita activa. La condizione umana*, Bompiani, Milano 1994 orig. 1958, p. 175. [24] Mastantuono, *La profezia straniera* cit., p. 23. [25] Ricœur, *La memoria, la storia, l'oblio* cit., p. 702.

FIGURE 29: A page from the journal Prospettiva Persona.

SAINT AUGUSTIN, SERMONS 293 ET 299 439

tur patris. Referte quod factum est ad significantem imaginem rerum ;
50 tamen quod factum est ne non factum putes, quoniam quid significaret forsitan discis. Hoc quod factum est, refer ad significationem rerum, et uide magnum mysterium : Zacharias tacet, amittit uocem, donec Iohannes nascatur praecursor Domini, et aperit uocem. Quid est silentium Zachariae, nisi prophetia latens et ante praedicationem
55 Christi quodammodo occulta et clausa ? Aperitur illius aduentu, clara fit uenturo eo qui prophetabatur. Hoc est apertio uocis Zachariae in natiuitate Iohannis, quod est discissio ueli in cruce Christi. Iohannes si seipsum annuntiaret, Zacharias os non aperiret. Soluitur lingua, quia nascitur uox. Nam Iohanni iam praenuntianti Dominum dictum est :

---

49 referte] *KN* $r^{5\,(a.c.)}$, refert $r^{1\text{-}4}$ p^5 t^2 *b vign*, referre $r^{5\,(p.c.)}$, refer *I* $p^{1\text{-}4}$ t^1 *maur* ‖ ad] ut r^4 ‖ significantem] -candam p^5 ‖49-50 rerum tamen] uerumtamen r^4, r. tantum *edd*

50 ne] *om.* r^5 ‖50-51 quid significaret] *N* $r^{1\text{-}3}$ *recc edd*, aliquid significare *K* $r^{4.\,5}$

51 forsitan] *om. K*, forsan *b vign* ‖ discis] *N*, perdiscis *K*, dicis $r^{1\text{-}4}$, dices *I* $p^{1\text{-}5.\,5}$ *t b edd*, disces p^4, alcius possit $r^{5\,(p.c.)}$, $r^{5\,(a.c.)}$ *non legitur* ‖ quod] *om.* $r^{1.\,3}$ ‖ refer] refert $r^{1.\,3.\,4\,(p.c.)}$, refertum est $r^{4\,(a.c.)}$

52 tacet] *KN* $r^{4.\,5}$, t. et $r^{1\text{-}3}$ *recc edd*

53 nascatur] *KN* r^4, nasceretur $r^{1\text{-}3}$ *recc edd, def.* r^5 ‖ aperit] *KN* $r^{2.\,5}$, -riuit r^1, -ruit r^3, -riat r^4, aperiret *recc edd* ‖ quid] r^4 *recc edd*, quod *KN* $r^{1\text{-}3}$, *def.* r^5

54 silentium zachariae] *inu.* p^5 ‖ zachariae] zache- t^2 ‖ nisi] ni r^3

55 christi] domini r^4 ‖ quodammodo] quodadmodo r^1 ‖ apertur] apertio *b* ‖ aduentu] -tum $r^{1.\,3}$, -tus r^4 *b* (p.c. ut uid.) ‖ clara] clausa r^4, et c. p^4

56 fit] fuit r^2 ‖ uenturo] et u. *I* $p^{2\text{-}4}$ ‖ hoc] haec r^5

57 discissio ueli] scissio u. *N*, dissisio u. r^4, ***** uelut t^2

58 seipsum] se ipse r^4 ‖ annuntiaret] *KN* $r^{4.\,5}$, nuntiaret $r^{1\text{-}3}$ *recc edd* ‖ zacharias] *KN* $r^{1.\,5\text{-}5}$, -riae r^2 *I p* t^1 *b edd*, zacherie t^2 ‖ non] nam *vign* ‖ soluitur lingua] *inu.* r^4

59 nascitur uox] *inu.* r^4 ‖ nam] non *vign* ‖ iam] *om. b vign*

---

52 « Vide magnum mysterium » : *s. Dolbeau* 3, 7.

52-53 cf. *Lc.* 1, 22, 64.

56-57 cf. *Lc.* 1, 64.

57 *Mt.* 27, 51 (*Mc.* 15, 38 ; *Lc.* 23, 45).

FIGURE 30: A François Dolbeau critical edition.

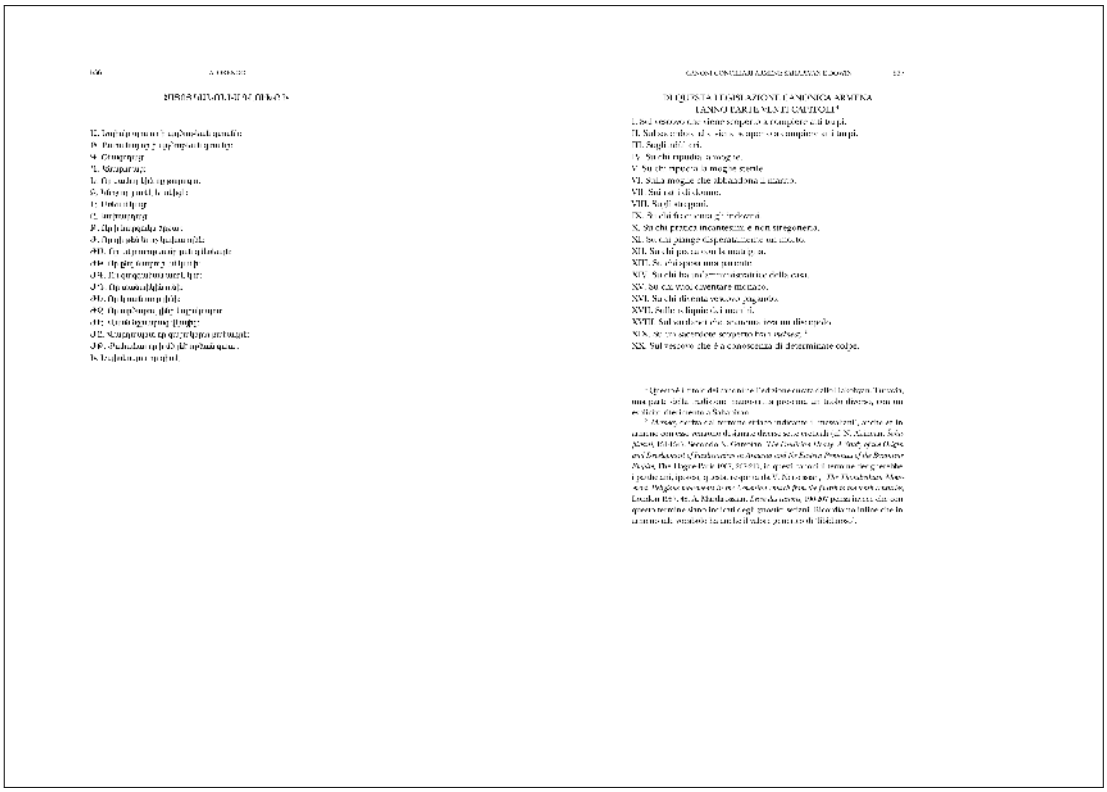


FIGURE 31: A parallel translation (Armenian-Italian) published in Augustinianum.

environment) with auto-completion, syntax highlighting, PDF visualization capabilities and much more functions.

Some other dedicated IDEs are Kile, TEX Works, TEXstudio, TEXnic Center. Users can configure them to decide what typesetting engine have to be used, to debug their documents and so on. The latest listed here, despite being a promising IDE, has been last updated back in 2014, which means that it has been dismissed.

Overleaf is a web site that allows users to create LATEX documents and collaborate on them. It offers an on-the-fly visualization of the resulting PDF.

The Wikipedia page WIKIPEDIA (2019) lists a lot of editors and users can try one or more of them to pick their favorite. The most part of them are intended to work on the source code. They are better than a simple text editor because their capabilities and facilities can help users to be more focused and productive. Some of them are WYSIWYM (what you see is what you mean): despite the document looks so much different from the final document, it shows the structure in a way clearer for the authors and they can try to visualize the final document in their mind, before viewing as an on-screen preview.

The only real WYSIWYG editor seems to be TEXmacs, a GNU program inspired by Emacs and TEX, but totally unrelated to them. It does not even use TEX to typeset the final document.

In case you do not have a favorite editor (for instance, ours are vi and Emacs) you may try some of them and decide which one is yours.

In the next section we present some highlights on LYX.

10 LYX, the WYSIWYG (?) Editor that LATEXs Your Documents

First of all, this section title is not completely true: the right title should have been "LYX, the WYSIsWYG..." i.e., the only "what you see is *sometimes* what you get". The Wikipedia page WIKIPEDIA (2019) states that LYX is properly a WYSIWYM editor.

Users who usually write their text documents with a WYSIWYG editor like Microsoft Word or LibreOffice Writer will find very hard to think of their documents the way LATEX requires. They feel comfortable, and possibly inspired, with an editor mimicking a white paper sheet. These users would probably be happy to use LYX, a program that looks very much like their favorite WP and that uses TEX as typesetting engine to get a LATEX document. They run LYX, see a nice user interface similar to those of the most famous WPs and... "Where is the white page?" Just a miserable yellowish band with a blinking cursor. Figure 32 shows LYX's new document. They do not need to won-

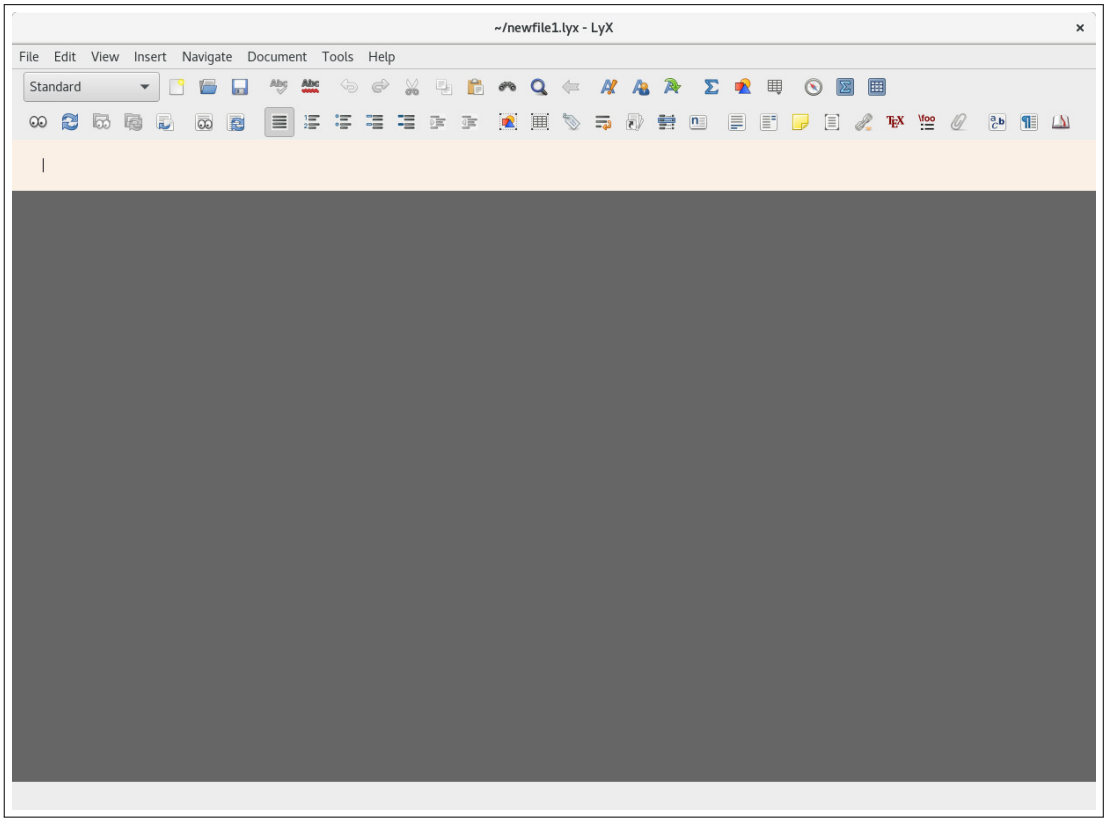


FIGURE 32: LyX and a new document.

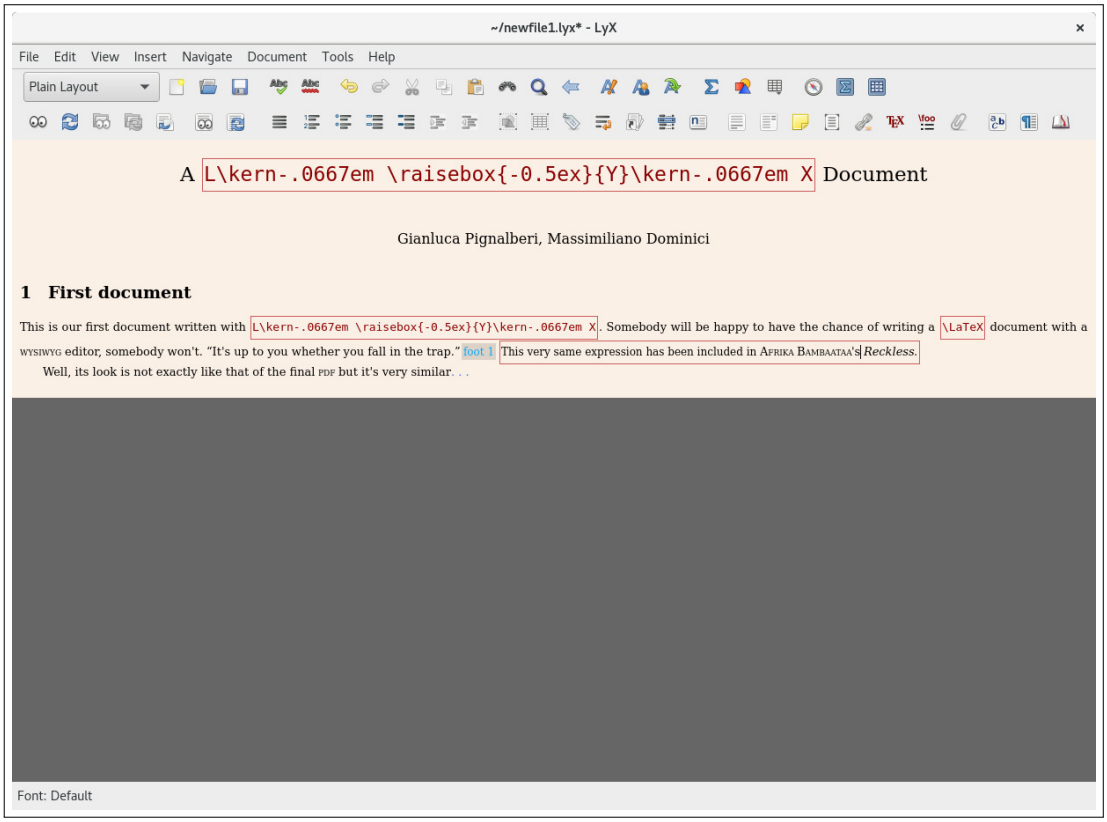
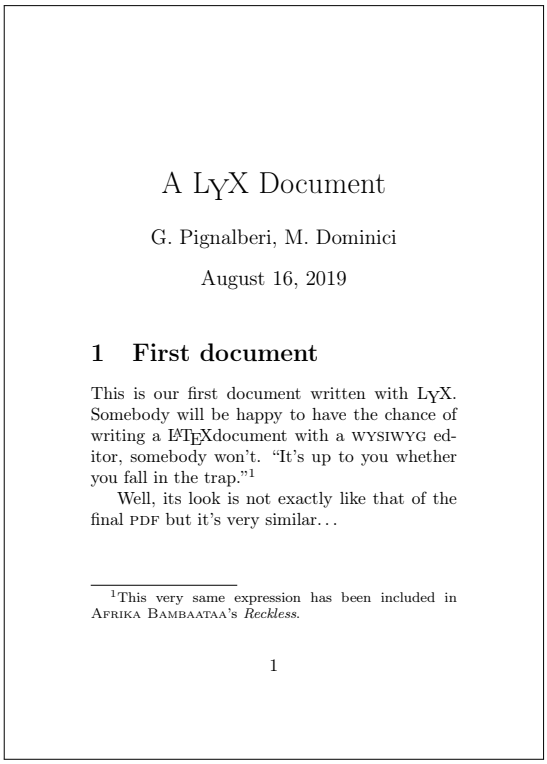


FIGURE 33: LyX and a new document.


A LyX Document

G. Pignalberi, M. Dominici

August 16, 2019

1 First document

This is our first document written with LyX. Somebody will be happy to have the chance of writing a LaTeXdocument with a WYSIWYG editor, somebody won't. "It's up to you whether you fall in the trap."[1]

Well, its look is not exactly like that of the final PDF but it's very similar...

[1]This very same expression has been included in Afrika Bambaataa's *Reckless*.

1

FIGURE 34: LyX and the new document exported in PDF.

der how will they write the whole text: the band enlarges as they keep writing.

A LyX file is not directly a LaTeX document, but it is very easy to export it in that format, so to post-produce it with LaTeX. Figure 33 shows that a document written into LyX only resembles the corresponding PDF (in figure 34), but the essence is there and that is the reason for Wikipedia (2019) to consider it WYSIWYM: we can inject TeX code into such a document to add unknown-to-LyX specific strings; we can "decide" with our mouse whether a portion of text is a title, a section, a footnote; we can apply emphasis or small caps (that the interface tags as "author" because some bibliography styles want the authors in small caps) to a portion of text highlighted with the mouse. LyX helps you manage a bibliography, lets you insert specific LaTeX features such as index entries, cross references and labels, tables of contents and many other elements.

In our basic example we used the default document class—article—with a A4 page size. Of course you can pick your needed type of document from Document→Settings... menu; you have a lot of things that you can customize there.

Click on the eyes to see the final PDF. When it suits your needs, you can export it with File→Export menu.

A Summary of textcomp commands

Apparently no manual groups textcomp commands all together.[16] We hope that our table 4, that shows all the textcomp symbols, helps you all in being more productive

You surely realized that two of the listed symbols are not visible. Those symbols, `\textascendercompwordmark` and `\textcapitalcompwordmark`, are "two additional compound word marks [...] that have the height of the ascender or capitals in the font, respectively." (Mittelbach and Goossens, 2013, p. 365). Those symbols are an extension of the LaTeX `\textcompwordmark`. They are zero-width characters (actually spaces) useful to prevent unwanted ligatures (do you remember the shelfful example? Try `shelf\textcompwordmark ful`) or to place an accent between two letters, as in the example of Mittelbach and Goossens (2013), the abbreviation of the German suffix -burg: `b\u\textcompwordmark g` and `B\u\textcapitalcompwordmark G`. In this case the zero-width symbol is the parameter of the accent command.

References

LaTeX 3 Project Team (2005). *LaTeX 2ε font selection*. Readable with `texdoc fntguide`.

Blanco, José Luis (2015). «Word or LaTeX typesetting: which one is more productive? Finally, scientifically assessed». Mapping Ignorance. https://mappingignorance.org/2015/04/06/word-or-latex-typesetting-which-one-is-more-productive-finally-scientifically-assessed/.

Bloch, Laurent (2017). «Efficacité comparée de latex et de ms-word». https://www.laurentbloch.net/MySpip3/Efficacite-comparee-de-LaTeX-et-de-MS-Word.

Burton, Tim (1997). *The Melancholy Death of Oyster Boy & Other Stories*. Rob Weisbach Books.

Carlisle, David, Scott Pakin and Alexander Holt (2001). *The Great, Big List of LaTeX Symbols*. https://www.rpi.edu/dept/arc/training/latex/LaTeX_symbols.pdf.

Charette, François (2015). *Polyglossia: An Alternative to Babel for XeLaTeX and LuaLaTeX*. Readable with `texdoc polyglossia`.

16. Well, we found an old manual (Carlisle *et al.*, 2001) that alphabetically groups textcomp commands and symbols into table 18. Some of those symbols are currently dismissed and a lot more have been added since then. Mittelbach and Goossens (2013, pp. 362–368) explains in details textcomp and table 7.6 summarizes its symbols.

TABLE 4: textcomp commands and symbols. We present them in the very same order of appearance in textcomp.sty.

COMMAND	=	COMMAND	=	COMMAND	=
\capitalcedilla	¸	\capitalogonek	˛	\capitalgrave	`
\capitalacute	´	\capitalcircumflex	^	\capitaltilde	~
\capitaldieresis	¨	\capitalhungarumlaut	˝	\capitalring	°
\capitalcaron	ˇ	\capitalbreve	˘	\capitalmacron	¯
\capitaldotaccent	˙	\textcapitalcompwordmark		\textascendercompwordmark	
\textquotestraightbase	‚	\textquotestraightdblbase	„	\texttwelveudash	‒
\textthreequartersemdash	—	\textdollar	$	\textquotesingle	'
\textasteriskcentered	*	\textfractionsolidus	/	\textminus	−
\textlbrackdbl	⟦	\textrbrackdbl	⟧	\textasciigrave	`
\texttildelow	~	\textasciibreve	˘	\textasciicaron	ˇ
\textgravedbl	“	\textacutedbl	″	\textdagger	†
\textdaggerdbl	‡	\textbardbl	‖	\textperthousand	‰
\textbullet	•	\textcelsius	℃	\textflorin	f
\texttrademark	™	\textcent	¢	\textsterling	£
\textyen	¥	\textbrokenbar	¦	\textsection	§
\textasciidieresis	¨	\textcopyright	©	\textordfeminine	ª
\textlnot	¬	\textregistered	®	\textasciimacron	¯
\textdegree	°	\textpm	±	\texttwosuperior	²
\textthreesuperior	³	\textasciiacute	´	\textmu	µ
\textparagraph	¶	\textperiodcentered	·	\textonesuperior	¹
\textordmasculine	º	\textonequarter	¼	\textonehalf	½
\textthreequarters	¾	\texttimes	×	\textdiv	÷
\texteuro	€	\textohm	Ω	\textestimated	℮
\textcurrency	¤	\capitaltie	‿	\newtie	‿
\capitalnewtie	‿	\textleftarrow	←	\textrightarrow	→
\textblank	␣	\textdblhyphen	=	\textzerooldstyle	0
\textoneoldstyle	1	\texttwooldstyle	2	\textthreeoldstyle	3
\textfouroldstyle	4	\textfiveoldstyle	5	\textsixoldstyle	6
\textsevenoldstyle	7	\texteightoldstyle	8	\textnineoldstyle	9
\textlangle	⟨	\textrangle	⟩	\textmho	℧
\textbigcircle	◯	\textuparrow	↑	\textdownarrow	↓
\textborn	★	\textdivorced	⚯	\textdied	†
\textleaf	🍃	\textmarried	⚭	\textmusicalnote	♪
\textdblhyphenchar	=	\textdollaroldstyle	$	\textcentoldstyle	¢
\textcolonmonetary	₡	\textwon	₩	\textnaira	₦
\textguarani	₲	\textpeso	₱	\textlira	£
\textrecipe	℞	\textinterrobang	‽	\textinterrobangdown	⸘
\textdong	₫	\textpertenthousand	‱	\textpilcrow	¶
\textbaht	฿	\textnumero	№	\textdiscount	⅋
\textopenbullet	○	\textservicemark	℠	\textlquill	⁅
\textrquill	⁆	\textcopyleft	℗	\textcircledP	Ⓟ
\textreferencemark	※	\textsurd	√	\textcircled	◯
		\t	‿		

David P. Carlisle and The LaTeX3 Project (2017). *Packages in the 'graphics' bundle*. Readable with `texdoc graphicx`.

Giacomelli, Roberto and Gianluca Pignalberi (2018). «Typesetting and highlighting Unicode source code with LaTeX: a package comparison». *ArsTEXnica*, (18), pp. 39–54.

Gregorio, Enrico (2010). «Installare TeX Live 2010 su Ubuntu». *ArsTEXnica*, (10), pp. 7–13.

Knauff, Markus and Jelica Nejasmic (2014). «An efficiency comparison of document preparation systems used in academic research and development». *PLoS ONE*, **9** (12), p. e115 069. https://doi.org/10.1371/journal.pone.0115069.

Knuth, Donald E. (1999). *Digital Typography*. Center for the Study of Language and Information, Stanford, CA.

Kopka, Helmut and Patrick W. Daly (2004). *Guide to LaTeX*. Addison Wesley, Boston, 4th edition.

Kühl, Philipp and Daniel Kirsch (2019). «Detexify latex handwritten symbol recognition». http://detexify.kirelabs.org/classify.html.

Lamport, Leslie (1987). MakeIndex*: An Index Processor for LaTeX*. Readable with `texdoc makeindex`.

Lavagnino, John (2003). *The endnotes package*. Readable with `texdoc endnotes`.

Mittelbach, Frank and Michel Goossens (2013). *The LaTeX companion*. Addison Wesley, Boston, 2nd edition.

Mittelbach, Frank, Robin Fairbairns, Werner Lemberg and LaTeX 3 Project Team (2016). *LaTeX 2ε font encodings*. Readable with `texdoc encguide`.

Oetiker, Tobias, Hubert Partl, Irene Hyna and Elisabeth Schlegl (2018). *The Not So Short Introduction to LaTeX 2ε*. A4 format, Version 6.2, February 28, 2018. Readable with `texdoc lshort`.

Pakin, Scott (2017). *The Comprehensive LaTeX Symbol List*. Readable with `texdoc symbols-a4`.

Powers, Shelley, Jerry Peek, Tim O'Reilly and Mike Loukides (2002). *Unix Power Tools*. O'Reilly, Sebastopol, CA.

Robbins, Arnold, Elbert Hannah and Linda Lamb (2008). *Learning the vi and Vim Editors*. O'Reilly, Sebastopol, 7th edition.

Robertson, Will and Khaled Hosny (2017). *The fontspec package*. Readable with `texdoc fontspec`.

Rooney, Garrett (2005). *Practical Subversion*. APress, Berkeley.

Schmidt, Walter (2006). «Font selection in LaTeX: The most frequently asked questions». *The PracTEX Journal*. https://www.tug.org/pracjourn/2006-1/schmidt/schmidt.pdf.

StackExchange (2011). «macros - Different command definitions with and without optional argument». https://tex.stackexchange.com/questions/308/different-command-definitions-with-and-without-optional-argument.

— (2016). «macros - What di the commands inside the LaTeX logo do?» https://tex.stackexchange.com/questions/313527/what-do-the-commands-inside-the-latex-logo-do.

Umeki, Hideo (2010). *The geometry package*. Readable with `texdoc geometry`.

Wikipedia (2019). «Comparison of TeX editors». https://en.wikipedia.org/wiki/Comparison_of_TeX_editors.

Bonus Section: Solution

You cannot pretend to read it that easy. Get a mirror! This is the 500th anniversary of Leonardo da Vinci's death.
All of the documents shown in figures 15–31 have been typeset by Massimiliano Dominici or Gianluca Pignalberi with LaTeX, being it pdfLaTeX, XeLaTeX or LuaLaTeX.

▷ Gianluca Pignalberi
g dot pignalberi at gmail dot com

▷ Massimiliano Dominici
mlgdominici at gmail dot com