Using `Typst` for Letters

R (Chandra) Chandrasekhar

2024-01-01 | 2024-01-01

Each of of us has had to write a letter to someone at some time in our lives. In the days of snail mail we used to write by hand, using lightweight paper, specially made for conveying the maximum information for the least weight of paper plus envelope. Those days are well and truly behind us now, to the extent that many of us rarely lift a pencil or put pen to paper, except to sign our names or perform some perfunctory calculation.

After the dedicated but computationally wasteful word processor was replaced by the general purpose personal computer, it was but a short hop, step, and jump for the PC to simulate a word processor among its innumerable masquerades, to satisfy our need to be connected.

The advent of the LaTeX typesetting system was yet another boon for those of us who laboured, with scissors and typed manuscripts, to publish papers in learned journals, especially in fields requiring mathematics for the exposition.

Nevertheless, I have always been dismayed by how cumbersome it has been to typeset a simple letter using LaTeX. One has to jump deftly through numerous hoops to get a decent-looking letter, even with the formidable resources of LaTeX.

So, I was with eager anticipation when I first stumbled upon mention of `Typst` [1,2] which exhorted "Compose papers/theses faster". The Typst home page claims it is a "A new markup-based typesetting system that is powerful and easy to learn." Piqued by these promising assertions, I decided to take the plunge with Typst for the specific but express purpose of writing letters. I have chronicled my experiences here.

The road to Typst

A glancing encounter with LaTeX3

I have been using LaTeX to write papers, theses, and books for well nigh thirty years. Thus, I am a beneficiary of LaTeX, and can vouch that the results from it are superior to those from a widely used proprietary "word processing" program which shall remain unnamed.

Despite that, I was somewhat aghast when I saw how the new, experimental LaTeX3 "dialect" or "macro language", or whatever it is called, looked like. It is the successor to the "LaTeX2ε" we use now, and while it might not impact end users much, it will certainly impinge upon customizability. The code fragment below below is taken from a blog entitled "LaTeX3: Programming in LaTeX with Ease" [3], and boy, did it make me uneasy, even to gaze upon it, let alone attempt to understand it:

```
\ExplSyntaxOn
% put the title in \l_tmpa_tl
\tl_set:Nn \l_tmpa_tl {My~Title}
% construct the command in \l_tmpb_tl
\tl_set:Nx \l_tmpb_tl {\exp_not:N \section* {\l_tmpa_tl}}
\cs_meaning:N \l_tmpb_tl % macro:->\section *{My Title}
% place the content of \l_tmpb_tl into the input stream
\tl_use:N \l_tmpb_tl
\ExplSyntaxOff
```

If you are as put off as I was by the code above, we are already on the same page. If you are a diehard LaTeX bhakt, you are unlikely to be swayed by what I have to say, and might as well stop reading now. If you are a fence-sitter, read along and make up your mind at the end.

The Pandoc-Markdown duo

By the time these blogs were written, I had already started using Pandoc and its extended Markdown in much of what I wrote. They are an efficient duo, and with the help of LaTeX, I have been able to produce a PDF document that bears more than a passing similarity to the HTML5 blog originally generated from Markdown using Pandoc.

So, all was smooth sailing, until I needed to write letters, and was that experience ugly! Suffice it to say that letter writing was not on the minds of most document typesetting designers when they ideated their creations.

The attractions of Typst

Like LaTeX, Typst is a markup-based programmable typesetting system. But it is fast and easy to use. Typst provides easy customizability because the programming syntax used in scripting is human-readable. Sensibly, it acts on objects that have meaning in the context of a document. It affects both the structure and content of a document. Like LaTeX, it produces PDF, which is bound to paper sizes. Should Typst also provide HTML output, it would have a killer-advantage over LaTeX, in addition to its current unique selling points (USPs).

I think of Typst as a domain-specific language, tuned to its purpose of typesetting documents, while remaining human-friendly. And because it is brash and young—even though it is a bit rough at the edges, regarding line breaks, paragraph breaks, and hyphenation—Typst will be sculpted to perfection in time, rather than be venerated as a sacred but untouchable relic, the way LaTeX now is. So, let us heave ho!

A short detour

Before wetting our feet on the beach of Typst with a letter, I want to showcase a small piece of boxed text that would have taken some time and effort to write in LaTeX, but which is a self-contained, compilable script in Typst.

Suppose we wanted the motto “Dharma protects those who protect it” in English, Sanskrit (संस्कृतम्), and Tamil (தமிழ்), within a coloured box with a border, we could do it so:


```
// dharma-blog.typ
// Compile with `typst compile dharma-blog.typ` to get PDF.

#rect(fill: silver, stroke: maroon,
text(fill: navy, size: 12pt, font: "Noto Serif",
"Dharma protects those who protect it."))

#rect(fill: silver, stroke: maroon,
text(fill: navy, size: 14pt, font: "Noto Serif Devanagari",
"□□□□□ □□□□□ □□□□□□"))

#rect(fill: silver, stroke: maroon,
text(fill: navy, size: 11.5pt, font: "Noto Serif Tamil",
"□□□□□ □□□ □□□□□□"))
```

“தர்மம் தலை காக்கும்”

I think that you would agree with me that the file to accomplish this multi-lingual, graphically enhanced text is about as concise as it can be, in order to be complete and self-contained. Moreover, the purpose of the text in the source file is easily comprehended, even by those unfamiliar with Typst. The result is shown in Figure 1. If you are incredulous enough to want to compile the source file for yourself, to get the PDF, here are the links for source and PDF.[1]

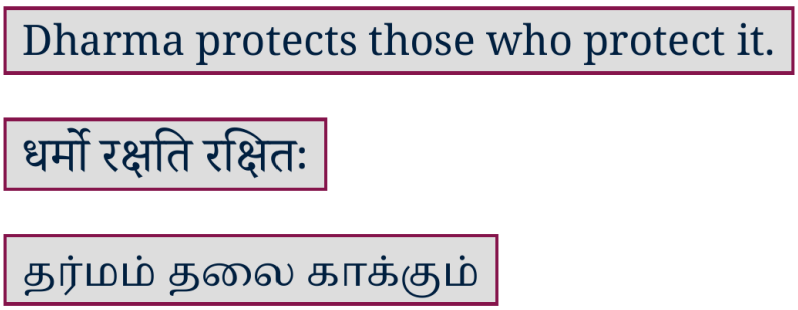

Figure 1: Output from the file `dharma-blog.typ` after compilation into a PDF.

Available letter templates in Typst

Let us now return to the main task of writing a letter using Typst. First, there are ready-made templates for letters in Typst that are available on the web. A cursory glance revealed that these conform to the DIN 5008 norm which introduces a degree of formality that is out of place in a social or casual letter. Moreover, developing a letter-template from scratch would reveal how simple or complicated a scripting language Typst is. That is the route taken here.

[1] Of course, you need the invoked fonts for successful compilation. If you are interested in other languages, whose fonts you do have, you could substitute texts in those languages and compile. To find out which fonts are seen by Typst, do `typst fonts`. I think you will agree that things could not get much simpler.

Sender's address at the top right

It is customary, when writing letters in English, to align the sender's address and the date to the top right of the page and for the recipient's address to be left-aligned, below that. Just these two text alignments used to take time and effort to achieve in LaTeX. And if you had a dual-signatory letter, things got even more entangled.

Let us see how Typst overcomes these hurdles and, in the process, appears almost too good to be true!

Toward a letter template in Typst

In LaTeX, we usually depend on packages written by others to do the medium-level lifting for our specific purposes, with simple option-like customizations left to ourselves. In Typst, we are ourselves empowered to customize our document, because template development[2] is not as forbidding a task as it is in LaTeX, which requires a steep learning curve.

The explanations on Typst that I give here might not be exact because I am a learner myself, and also because Typst itself is evolving. As I understand it, Typst allows content and its formatting to be specified in a template that marries the two. This is because the document we are creating may be structured into the natural components we envisage. In this sense, *Typst is a document-templating language*.

Once the template is complete, we invoke it much like a `\documentclass` command in LaTeX. But there the similarity ends. After invoking the template, we fill the content for each structural element of the document and we produce an *instantiation* of a letter from the template. Here, *Typst is a document-markup and typesetting language*. We then *compile* the letter into a PDF document and view the end result.

If the appearance of any element is amiss in the PDF, we can correct it at once in the source document and view the result once more. Because compilation is very fast, we can work interactively in the manner to get decent results in a short time.

Template reflects structure

The text below is the beginning of the template and embodies the document structure of a letter:

It has the same sequential spatial structure as a letter, starting at the top and moving to the bottom. All the defined fields have self-explanatory names. The content of the fields are set to `none` because this is a template rather than an instance of a letter. The actual values will be set in the specific letter we choose to write. The document template is therefore a *mapping* or set of key-value pairs in which the value is initialized to `none`. If we want the template to assume default values for unchanging fields like `from_name`, `from_address`, etc., we may set those in the template itself.

Note that at this stage, nothing has been said about page size, text alignment, font name, etc. All that follows *after* this structure has been laid out.

[2] Think `documentclass` in LaTeX.


```
#let letter_template(
  from_name: none,
  from_address: none,
  date: none,
  to_name: none,
  to_address: none,
  subject: none,
  salutation: none,
  content: none,
  closing: none,
  signatures: (),
  enclosures: (),
  cc: none,
  figures: (),
  footer: ()
  ) = {
```

Figure 2: Top of letter template showing the structure of the letter as fields.

Filling the template with content

When the fields in the template are filled in with content or data, we have the letter materializing out of the template. The markup, alignment, etc., are managed by the template. The completed letter is then passed through Typst with the simple command `typst compile <filename.typ>` we get as output the PDF of the letter we desire.

The markup, such as alignment, comes later in the template in the form of code that explains the markup associated with each element, as shown in Figure 4:

Note that content appears as strings within double quotes, or within (square) brackets. Because a typst file accommodates both a programming language and some content in the same file, we need to adhere to an unambiguous syntax to discriminate between code and content. The latter is shown within bracket delimiters `[...]` or within double quotes `"..."`.

Three files are available:

1. ‘[

Storm-damaged oak tree. Thanks to user 127071 at Pixabay.

Acknowledgements

I am deeply grateful to my son, Nandakumar Chandrasekhar, for constructing a generalized letter template in Typst at short notice, even though he did not have prior familiarity with the language.


```
#import "letter-template.typ": letter_template
#show: my_letter => letter_template(
  from_name: "The Dimbleby Family",
  from_address: "The Lodge
    Cheswick Village
    Middle Upton
    Bristol BS16 1GU",
  date: "31 December 2023",  // Date will be displayed as is.
  to_name: "Evergreen Tree Surgeons",
  to_address: "Midtown Lane
    Cheswick Village
    Stoke Gifford
    Bristol BS16 1GU",
  salutation: "Gentlemen,",
  subject: "Pruning of Heritage Oak Trees in the Dimbleby Estate",
  content: my_letter,
  closing: "Sincerely yours,",
```

Figure 3: Top of actual letter with content filled in.

```
set align(right)
from_name
linebreak()
from_address

set align(right)
date

set align(left)
to_name
linebreak()
to_address
```

Figure 4: Code for aligning the `from_name` and from_address`to the right and` theto_name`and`to_addressee‘ to the left.

Feedback

Please email me your comments and corrections.

A PDF version of this article is available for download here:

https://swanlotus.netlify.app/blogs/using-typst-for-letters.pdf

https://www.alanshawn.com/tech/2020/10/04/latex3-tutorial.html

References

[1] reknih. 2023. Typst, a new markup-based typesetting system, is now open source. Retrieved 1 January 2024 from https://news.ycombinator.com/item?id=35250210

[2] desperado339. 2023. Check out Typst, a modern LaTeX alternative written in Rust. Retrieved 1 January 2024 from https://www.reddit.com/r/rust/comments/18001dr/check_out_typst_a_modern_latex_alternative/

[3] Alan Xiang. 2020. LaTeX3: Programming in LaTeX with Ease. Retrieved 1 January 2024 from https://www.alanshawn.com/tech/2020/10/04/latex3-tutorial.html

