learnlatex.org: Taking LaTeX training fully interactive

David Carlisle, Paulo Roberto Massa Cereda, Joseph Wright

1 Introduction

There are a plethora of resources available to new LaTeX users. However, it is much more difficult to discover which of these provides the best introduction to LaTeX. These online resources vary in quality and correctness: over time, and with limited editing, even good advice can become out-of-date. Many good resources are over-detailed for a new user who needs only straightforward help to get over the initial barrier to using the system.

For many programming languages there are now web sites providing the opportunity to try coding online using a cloud compiler. These cloud compilers can be harnessed by a range of teaching websites to offer a simple introduction to the language using a suitable IDE (integrated development environment); a good example is `LearnPython.org`. Such sites tend to be limited in scope as they are not aiming to teach every possible idea in the language but only a "Beginners" menu.

Over the past six months, work has been ongoing in filling that gap for LaTeX: `learnlatex.org`. The aim of this new site is to provide a carefully-curated set of resources for beginner LaTeX authors, with integrated use of an online LaTeX environment and demonstrations accessed directly from these lessons. The scope of these learning resources is focussed, and with the aim of offering the material in bite-sized chunks.

2 Existing resources

There are already many online resources for learning LaTeX, as a simple web search will reveal. The top hit at present is Overleaf's "Learn LaTeX in 30 Minutes" (`https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes`), which covers many of the core ideas with a series of examples. The second is `https://www.latex-tutorial.com`, which takes the step of dividing up the lessons into a series of pages. There are then more "traditional" resources, including the long-standing LaTeX Wikibook (`https://en.wikibooks.org/wiki/LaTeX`), and of course many excellent printed LaTeX guides.

The Overleaf page is notable as Overleaf provides a full LaTeX system online, and is used by many people as their LaTeX editor/system. On their teaching page, the examples are given as code blocks but to run then, a separate tab needs to be opened. That has the positive of looking exactly the same as any other document edited on Overleaf, but means leaving the teaching page. The other leading hits for learning LaTeX require that the learner copies or types out the examples by hand.

In contrast, users learning other programming languages can today often start on websites that let them try out the system *in the page*. That means no copy-pasting, no need to move to other tabs, and no need to download and install software. As mentioned above, `LearnPython.org` is an excellent example, forming part of a family of around half a dozen sites using the same overall framework to teach a set of modern languages.

The `learnlatex.org` project was started with the aim of combining existing best-practice teaching on LaTeX with a strong focus on interactive, online, training. That required tackling three things: the content, the website structure, and the online element.

3 Writing content

The starting point for writing the content was long-standing material developed by Nicola Talbot and used by the UK TeX User Group (UK-TUG) for face-to-face lessons (`https://github.com/uktug/latex-beginners-course`). After assembling a small group of volunteers, we began by looking at this curriculum, the content of other sites and discussions we have all had with new(er) users. That led to a first set of section titles, which we adjusted as new ideas arose.

We then started drafting the content, taking existing notes plus new material to deliver around 15 lessons.[1] As we worked through these, we found we had more to say than was reasonable to cover in a focussed set of lessons. We also felt there was good content that did not belong in the basic lessons, but did belong somewhere. That led to a split of each lesson into two parts: the basics and "going further". (We'll look at the file structures below, and how they build the site.)

As well as the content and demonstrations, we've worked hard to make sure that the lessons are up-to-date and accurate. That's led to discussions with the LaTeX team about for example `\label`, which is reflected in the lesson content and means that we are confident the site covers *best practice*.

4 Website structure

Based on experience with `texfaq.org`, we knew that GitHub Pages, which lets you write webpages in Markdown, was a good place to look to host the site.

[1] We've grown to 16 lessons, as there was a strong argument to add one on errors.

This means that multiple authors can easily work with the sources without needing to set up a complex build system: the pages can be edited on the GitHub site, so a local Git installation is not even required.

We started with a simple structure, with an index and 15 `lesson-XX` files (all in Markdown, of course). GitHub Pages runs a system called Jekyll, which turns those files into the full site. That's done using various bits of template plus some scripting, which can all be user controlled. As the site has grown, we split the extra content into `more-XX` files, which can then be linked automatically to their "parent".

Not long after we started work, the question of translations came up. We'd started with the idea of just working in English, but we didn't want to box ourselves in. So we moved all of the content into a directory called `/en`, and started exploring how to add a language selector. To help with that, we made some stub files that were marked as being in a few languages, then got at least the page titles translated by real people. It turns out to be reasonably easy to add a selector, and there's already been interest in translation, so we have the site in English, Vietnamese and Portuguese already, with Spanish, German and Japanese in the pipeline.

Linked to translations, we also realised that while most of the lessons stay the same for every language, we needed a place where language-specifics could be covered. That came up when we were asked about Japanese; they need vertical typesetting, which of course is very different from what we need for English. So a place for non-translated pages has been added: the content and number of pages there is down to the translators.

5 Online LaTeX compilers

As outlined above, a key aim in developing the site has been to allow examples to be run as close as possible to "in the page": that has led us to explore a range of options in this area. We wanted to have a full LaTeX system available, which mean that systems such as MathJax (`mathjax.org`) or even MiniLaTeX (`minilatex.app`) were not suitable.

5.1 Processing examples "in the page"

It is possible to run TeX in the JavaScript engine in the browser and initially we did some experiments with `texlive.js` but such systems are hard to keep up to date and currently provide only pdfLaTeX, not other engines such as LuaLaTeX. So the decision was made to use a server which ran LaTeX, returning the generated PDF to the web browser. An important consideration when choosing the technology here was that the examples should cover a range of languages and so a range of TeX engines should be available, notably pLaTeX, LuaLaTeX and XeLaTeX in addition to pdfLaTeX.

During initial development of the site, we mostly used the servers `latexonline.cc` and latex-on-http (`https://latex.ytotech.com/`). However, in the end we developed a server `latexcgi.xyz` specifically tailored to the requirements of the site.

The most noticeable distinguishing feature of the LaTeX CGI server is that it does not directly return the generated PDF but instead returns a call to a locally hosted copy of the `PDF.js` (`https://mozilla.github.io/pdf.js`) JavaScript library. This ensures consistent behaviour, notably on mobile browsers which typically do not include a built-in PDF renderer. The server does support options to return the PDF directly, or to return the log file, even of successful runs.

LaTeX CGI supports a range of LaTeX formats: `lualatex`, `pdflatex`, `xelatex`, `uplatex`, `platex` and their `-dev` variants. It also supports `biber`, `bibtex`, `pbibtex` and `bibtex8` and `makeindex`.

The server has installed a full copy of TeX Live 2020 and runs in the default "restricted shell escape" mode. This means that packages such as `imakeidx` that make use of the allowed shell commands are available.

Currently `latexcgi.xyz` is hosted as a virtual machine on Amazon's EC2 service, which is available in the *Free Tier* for one year. Long term hosting has yet to be finalized.

Using a dedicated server gives some flexibility in the technologies used. One possibility would be to use Docker images such as the TeX Live images provided by Island of TeX (`https://gitlab.com/islandoftex/images`). This would likely be necessary if we decide to offer more programming features, in particular allow TeX to be run with shell escape enabled, although currently the intention is to match the default (restricted shell escape) that matches a typical user's system.

5.2 Online TeX processing systems

All the systems mentioned in the previous section have the feature that they require no login or pre-registration, which is convenient but means that any edits to the documents are lost when the user moves off the page. The site therefore offers a second alternative to access a full online TeX system provider, currently Overleaf.

If the **Open in Overleaf** button is used, a new project is opened in Overleaf in a new tab in the browser. The user can edit the example in Overleaf, and save the project if desired, to return to it later.

The Overleaf API was extended for this site, to allow multiple-file projects to be uploaded in this way.

Other TEX editing systems could be used but a requirement is a public API that starts a project via an HTTP request rather than using the web page menu interface.

6 What's next

The web design we are using is very basic: just what comes out-of-the-box with GitHub Pages. To help users navigate, we need proper development of a full site. That's beyond the expertise of the site team, but we have been able to raise funds to employ a professional web developer. He's looking at design both in terms of appearance and structure: things like mobile accessibility, metadata, etc., are all really important.

There's also still more to do on the content, in particular working out if we need more formalised exercises or lesson summaries. That likely needs user feedback, for which we need *your* help. The site will work best if it's promoted to potential users, and readers of *TUGboat* are we hope well-placed to recommend it. Of course, to do that, we'd encourage you to read the site, give feedback and perhaps even suggest some improvements!

◇ David Carlisle
Oxford, United Kingdom
`d dot p dot carlisle (at) gmail.com`

◇ Paulo Roberto Massa Cereda
São Paulo, Brazil
`paulocereda (at) gmail.com`

◇ Joseph Wright
Northampton, United Kingdom
`joseph dot wright (at) morningstar2.co.uk`