TUD Jupyter Book Templates

Freek Pols

1. Introduction to Jupyter Book

Jupyter Book is an open-source tool designed to create interactive teaching materials, such as textbooks and tutorials, using Markdown, Jupyter Notebooks, and other formats. It enables educators to combine narrative text, executable code, and multimedia content in a single, shareable resource.

> **Warning**
>
> Once started with Jupyter Books, you might get hooked!

There are two main versions: JB1 (the original) and JB2 (the current, more feature-rich version). Both support integration with tools like Visual Studio Code (VSC) for editing, Git for version control and collaboration, and Python for interactive code examples.

At Delft University of Technology, we actively support the creation and publication of Jupyter Books, providing guidance and infrastructure for both authors and readers. The following chapters will explore these tools and workflows in more detail.

> **Try the Python code below**
>
> Click the ON-button at the top right and subsequently the play button to see the output of the Python code below.

Published Sep 09, 2025

2. Making your first interactive textbook

- think about your plan
- is it a self-contained educational textbook (OIT) or is it more of a reader (OCW)?
- do you want to publish it open access (with what restrictions, if any?)
- what is your level of expertise in terms of python, markdown, jupyter, git? This determines whether it is better to build on your local computer of online.
- what do you need help with?
- do you want to test the materials with students, so behind sso?

> **Warning**
>
> Under development

- intake
- project with support
- regular meetings
- copyright check
- plagiarism scan
- online check behind sso on OIT/dev OCW/dev
- final adjustments
- tag / release 1.0

3. Git

Git is a version control system that helps you track changes in your files and collaborate with others. It allows you to save different versions of your work, go back to previous versions if needed, and merge changes from multiple people. Git is commonly used for managing source code in software projects, but it can be used for any type of file.

Git provides thus a way to synchronize files between a server and local stored files. Other servers can host the webpage (the Jupyter Book) by requesting the files and utilizing a build script.

Both GitHub and GitLab provide an online integrated development environment: an online version of VSC. This approach (coding in the online IDE) is advised for those new to git, python, and programming.

Below is a description of both GitHub and GitLab - TU Delft and subsequently an oversight of the (dis)advantages of each these systems. Know that we are able to connect these two (synchronize between them).

3.a. GitHub

GitHub is a web-based platform for hosting and collaborating on Git repositories. Owned by Microsoft, GitHub offers tools for code review, issue tracking, and project management, making it popular for open-source and private projects alike. One valuable feature is GitHub Pages, which allows you to publish static websites directly from a repository, making it easy to share documentation, portfolios, or project pages.

3.b. Gitlab

TU Delft hosts a local server with GitLab software, which is an implementation of git with a lot of useful extensions. Anyone with a TU Delft netid can get a GitLab account, simply by logging in on https://gitlab.tudelft.nl. You can then be added to projects or start your own project.[1]

3.c. Git desktop

GitHub Desktop is a graphical application that simplifies working with Git repositories, making it easier to synchronize your files with platforms like GitHub or GitLab. Instead of using command-line instructions, you can perform common tasks—such as committing changes, pushing to remote repositories, and resolving merge conflicts—through an intuitive interface. This is especially helpful for beginners or those who prefer a visual workflow.

Alternatively, Visual Studio Code (VSC) includes built-in Git functionality. With VSC, you can stage, commit, and push changes directly from the editor, view diffs, and manage branches without leaving your coding environment. This integration streamlines your workflow and reduces the need to switch between multiple tools.

3.d. Comparing the two versions of git

[1] Text from Idema (2023)

GitHub and GitLab have both advantages and disadvantages. GitLab is advised by TUD unless many external users are involved. However, as GitLab disable pages a Jupyter Book should be build locally.

	Positives	Negatives
Github	GitHub pages enables to see content in website	Is property of MicroSoft
	Allows users out side TUD to easily engage	No backup at TUD
	Github actions	Not standard reachable by TUD (i.c.o. conflicts, e.g. copyrights)
	Massive storage	
Gitlab	Is hosted by TUD	Is terminated when one leaves TUD
	Accessible by colleagues	Harder for outside collaborators
	CI/CD	Pages disabled
	Storage limited by TUD	Various faculties have their own GL and policies
	TUD Backups	Is more often down compared to GitHub

4. Which template should I use?

We offer three templates, each with its own (dis)advantages. We specify below the characteristics in order to help you pick the 'best' template that fits your needs and level of expertise.

All templates provide the same starting chapters, telling a little about the markdown language, and providing some exercises to help you get started.

4.a. Jupyter Demo Book

The Jupyter Demo Book template (adapted from the original Jupyter Book templated by Timon Idema) is based on the Jupyter Book 1 software and utilizes TU Delft Gitlab. Full software installation is required to produce an output.

Positives:

- bare bone version
- software developed and support by JB1 developers

Negatives:

- Full software installation to see output
- No full control unless coded yourself
- Difficult to set up
- OIT support needed to get started
- PDF export suboptimal

Example book made utilizing the template: A Brief Introduction to Biochemistry: Biochemistry for non-chemists

4.a.i. *Requirements:*

- VSC
    - Extensions
        - Jupyter
        - Markdown
        - Github Actions
        - Python
- Python
    - Dependencies
        - jupyter-book
        - numpy
        - scipy
        - matplotlib
        - plotly==5.24.1
        - myst_nb

    - jupyterquiz
    - sphinx-exercise
    - sphinx-proof
    - pydata-sphinx-theme
- Gitlab

4.b. Teachbooks

The Teachbooks template (developed by Tom van Woudenberg, Robert Lanzafame and Dennis den Ouden-van der Horst) is based on the Jupyter Book 1 software and utilizes Microsoft Github. It is recommended to install full software to run the book locally, but is not required through the use of GitHub Pages.

Positives:

- Quick start
- Additional functionality built by teachers
- Full customizable through CSS

Negatives:

- Understanding all possibilities and functionalities
- Connecting Github and Gitlab to comply with TUD publishing regulations (but possible)
- PDF export suboptimal
- Public from start
- may feel as black box approach (what is under the hood)

Example book made utilizing the template: Linear Algebra

4.b.i. *Requirements:*

- Github

optional:

- VSC
    - Extensions
        - Jupyter
        - Markdown
        - Github Actions
        - Python
- Python
    - Dependencies
        - Teachbooks
        - git+https://github.com/TeachBooks/TeachBooks-Favourites
        - sphinx-tudelft-theme

4.c. Jupyter Book 2

The Jupyter Book 2 template (under development by Freek Pols) is based on the Jupyter Book 2 software and utilizes Microsoft GitHub.

Positives:

- High quality pdf with ease
- Newest technology, still in development
- Used in science publication

Negatives:

- Not all functionality from JB1 is available (yet)
- In browser Python is not editable
- Not full control of style
- Public from start if not built locally

4.c.i. *Requirements:*

- Github

optional:

- VSC
    - Extensions
        - Jupyter
        - Markdown
        - Github Actions
        - Python
        - MyST-Markdown
- Python
    - Dependencies
        - mystmd

Example book made utilizing the template: Introducing Classical Mechanics & Special Relativity

4.d. Which one than?

The choice for any of these templates depends on the user. If you want to engage readers with python code which is editable in the browser, don't choose JB2 (yet). If you also want to make a high quality LaTeX or Typst pdf, go for JB2. If you want to understand the whole JB technology and if you are very keen what to use (dependencies) go for the Jupyter Demo Book. If you are okay with a kind of black box where everything is done already for you, if you have hardly any experience with VSC, git and Python, go for the Teachbooks template. Below summarized:

Teachbooks	Jupyter Demo Book	Jupyter Book 2
JB 1	JB 1	JB 2
Customized sphinx extension	Traditional Jupyter packages	JS plugins
Full control o / adaptable / customizable layout through extensions	Full control through own code and picking dependencies	Less control
PDF from script	PDF from print	PDf from Typst / LaTeX conversion

5. To do / decide on

The three templates differ from the extend of information given. Ideally the three templates are more/well aligned and all comply with TUD regulations and huisstijl. Moreover, they should contain information on the author's journey, copyrights, plagiarism and a useful help by providing a well organized cheat sheet. To what extend do we want authors to choose their own format/style (deviating from TUD huisstijl)?

GitHub and GitLab have both their merits and trade offs. I guess we can allow both given a few boundary conditions. Do we want 'automatically' a copy to gitlab (if possible at all to make that mandatory)?

References

Idema, T. (2023). *Jupyter Open Textbook: demo book*. TU Delft OPEN Publishing. https://doi.org/10.59490/tb.73