Creating accessible and interactive documents with PreTeXt

A JMM2023 Professional Enhancement Program

Creating accessible and interactive documents with PreTeXt

A JMM2023 Professional Enhancement Program

Steven Clontz
University of South Alabama

Oscar Levin
University of Northern Colorado

2023 January

Preface

Note. This JMM Professional Enhancement Program will be held on Thursday, January 5 between 9 am - 11 am and Saturday, January 7 between 9 am - 11 am in Salon CD on the fourth floor of the Boston Marriott Copley Place.

PreTeXt[1] is a document authoring system that allows you to convert the source of your document into a variety of output formats, including fully accessible webpages, PDF, Epub, Jupyter Notebooks, and braille. This "write once, read anywhere" approach has made it a popular choice for authors of Open Educational Resources, but PreTeXt can also be used to create other kinds of mathematical documents as well. Recent updates make this process much easier; there has never been a better time to get started with PreTeXt.

Participants of this PEP will be introduced to the fundamentals of authoring documents with PreTeXt and gain the technical skills required to work with it. Specifically, participants will learn how to:

- Use GitHub Codespaces to create an editable PreTeXt document in their web browser.
- Write and structure content using PreTeXt markup.
- Add content to the document, including mathematics, graphics, interactive exercises, and more.
- Build accessible and interactive webpages as well as a static PDF from the same PreTeXt source.
- Easily deploy the interactive webpages online (for free).

We will also share tips for converting existing documents into PreTeXt.

Prior to the PEP, participants should have some familiarity with LaTeX or other markup languages. No previous experience working with PreTeXt or HTML/XML is assumed. Participants should bring a laptop that can connect to JMM's provided wifi: no prior installations will be required as we will use PreTeXt's new GitHub Codespace-powered online authoring service.

[1] `pretextbook.org`

Contents

Preface iv

1 Before you arrive... 1

1.1 Setting up your GitHub account . . . 1
1.2 Apply for your GitHub Education discount . . . 1
1.3 And that's it! . . . 1

2 Write Once, Read Anywhere 2

2.1 Setting up Codespaces . . . 2
2.2 PreTeXt Principles . . . 2
2.3 PreTeXt is XML . . . 3
2.4 Books and Divisions . . . 3
2.5 Paragraphs, Lists, and Blocks . . . 6
2.6 Figures and Diagrams . . . 7
2.7 Interactives . . . 9
2.8 Authoring an Exercise . . . 10
2.9 Not Just HTML, Not Just PDF . . . 11
2.10 Wrapping Up . . . 12

3 Codespace Workflow 13

3.1 Build and Generate . . . 13
3.2 Previewing . . . 14
3.3 Saving your work . . . 14
3.4 Deploy . . . 15

Appendices

A Getting Help 16

B Copyright and Licensing 18

C Acknowledgement 19

Chapter 1

Before you arrive...

Authoring in PreTeXt requires nothing more than a wifi connection and GitHub account. GitHub users also can share their content for free on GitHub Pages!

(The program description references CoCalc.com, another service that can be used to author PreTeXt, but workshop plans have been updated to focus primarily on GitHub workflows.)

Wifi will be available in our room at the JMM, so each participant will only need to bring their laptop. No installation of software is required

1.1 Setting up your GitHub account

To create a GitHub account, follow the instructions on GitHub's signup page.[1]

Be sure to note your GitHub username and password in your password manager (or however you usually keep track of login credentials) so you can log in again at the Joint Mathematics Meetings.

1.2 Apply for your GitHub Education discount

Educators and non-profit researchers can get many of GitHub's paid features for free.

Apply at Education.GitHub.com[1] to unlock these features (in our experience, applications are usually processed quickly for `.edu` email addresses).

1.3 And that's it!

Even just a couple months ago, the process to get started writing PreTeXt involved several more steps. (Raise your hand if you don't care what a "PATH variable" is!)

The community is streamlining this experience daily, and we look forward to sharing how easy writing in PreTeXt is when we meet you at the JMM!

[1] `github.com/signup`

[1] `education.github.com/discount_requests/pack_application`

Chapter 2

Write Once, Read Anywhere

Objectives

At the end of this chapter, you'll

1. Become aware of the eleven PreTeXt Principles.
2. Be able to identify several features of PreTeXt.
3. Have a working GitHub Codespaces environment to suitable for authoring and editing in PreTeXt.

2.1 Setting up Codespaces

A **Codespace** is an authoring environment that lives in the "cloud", that is, a virtual machine hosted by GitHub that has all of the software needed to create great accessible documents, accessible with just a web browser.

This coding environment uses a web version of Virtual Studio Code, an open-source editor, along with the PreTeXt community's custom plugins and software to get started authoring quickly.

Follow the instructions at `https://github.com/PreTeXtBook/pretext-codespace` to get started. Let this run for a few minutes in the background while you review the rest of this section.

2.2 PreTeXt Principles

A more detailed monograph on PreTeXt's philosophy[1] is availabe in the PreTeXt Guide.

List 2.2.1 PreTeXt Principles

1. PreTeXt is a markup language that captures the structure of textbooks and research papers.
2. PreTeXt is human-readable and human-writable.
3. PreTeXt documents serve as a single source which can be easily converted to multiple other formats, current and future.

[1] `pretextbook.org/doc/guide/html/philosophy.html`

4. PreTeXt respects the good design practices which have been developed over the past centuries.
5. PreTeXt makes it easy for authors to implement features which are both common and reasonable.
6. PreTeXt supports online documents which make use of the full capabilities of the Web.
7. PreTeXt output is styled by selecting from a list of available templates, relieving the author of the burden involved in micromanaging the output format.
8. PreTeXt is free: the software is available at no cost, with an open license. The use of PreTeXt does not impose any constraints on documents prepared with the system.
9. PreTeXt is not a closed system: documents can be converted to LaTeX and then developed using standard LaTeX tools.
10. PreTeXt recognizes that scholarly documents involve the interaction of authors, publishers, scholars, curators, instructors, students, and readers, with each group having its own needs and goals.
11. PreTeXt recognizes the inherent value in producing material that is accessible to everyone.

2.3 PreTeXt is XML

Since PreTeXt uses the XML markup language, all content is structured in terms of **elements**. The root `pretext` element nests many other elements inside of it. This is accomplished by surrounding everything with a starting **`<pretext>` tag** and an ending `</pretext>` tag. (Folks with HTML experience will find this pattern familiar, akin to the "HTML" root element.)

Listing 2.3.1 is a very simple PreTeXt/XML document. (The first line is boilerplate that lets various programs know the rest of the file is XML, and the third-to-last line is an example of a comment that won't appear in the output.)

```
<?xml version="1.0" encoding="UTF-8"?>
<pretext>
    <article>
        <title>Hello world!</title>
        <p>Welcome to PreTeXt!</p>
        <!-- TODO: find something more to say... -->
    </article>
</pretext>
```

Listing 2.3.1 Source of a simple PreTeXt book project.

2.4 Books and Divisions

There are several documents you can write in PreTeXt, such as `<article>`s and `<slideshow>`s. This tutorial will focus on `<book>`s.

A `<book>` typically includes `<frontmatter>`, and `<backmatter>`.

Between <frontmatter>, and <backmatter> are either several <chapter>s or <part>s. If used, <part>s are subdivided into <chapter>s. Then <chapter>s subdivide into <section>s, and <section>s can have <subsection>s.

Each of these subdivisions needs a <title>, and may have an <introduction> or <conclusion>.

Listing 2.4.1 puts some of these elements together for a simple PreTeXt project (information on the other elements will come in later sections).

```
<?xml version="1.0" encoding="UTF-8"?>
<pretext xml:lang="en-US">
  <!-- (author configurations go in docinfo) -->
  <docinfo>
    <macros>
      \newcommand{\R}{\mathbb R}
    </macros>
  </docinfo>


  <book xml:id="my-great-book">
    <title>My Great Book</title>
    <subtitle>An example to get you started</subtitle>

    <frontmatter xml:id="frontmatter">
      <titlepage>
        <author>
          <personname>You</personname>
          <department>Your department</department>
          <institution>Your institution</institution>
        </author>
        <date>
          <today />
        </date>
      </titlepage>
    </frontmatter>

    <chapter xml:id="chapter-welcome">
      <title>Welcome!</title>
      <introduction>
        <p>This chapter is about the real numbers
            <m>\R</m></p>
      </introduction>

      <section xml:id="section-getting-started">
        <title>Let's get started</title>
        <p>Can you solve <m>ax^2+bx+c=0</m>?</p>
      </section>

      <section xml:id="section-learning-more">
        <title>But wait, there's more!</title>
        <p>Did you know that
            <me>x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}</me>?</p>
      </section>

    </chapter>

    <backmatter xml:id="backmatter">
      <title>Backmatter</title>

      <colophon>
        <p> This book was authored in <pretext />. </p>
      </colophon>

    </backmatter>

  </book>

</pretext>
```

Listing 2.4.1 Source of a simple PreTeXt book project.

2.5 Paragraphs, Lists, and Blocks

Within each division (chapter, section, etc., see Section 2.4) of your book, you likely want some content (e.g. what you're reading right now!).

Written content is usually structured as **paragraphs**, `<p>` for short. If you've ever written HTML, this tag may be familiar to you, but be warned: while PreTeXt is XML (Section 2.3), *PreTeXt is not HTML*! There is some overlap: you can *emphasize* words or phrases with `<em>` for instance. However, while HTML uses the full word "code" for its tag, PreTeXt uses the shortened `<c>` tag.

Note that these elements are all **semantic**: they express the *meaning* of content, not its presentation. For example, the word "semantic" was a `<term>` we just defined, while we merely emphasized "meaning" with `<em>`. The presentation of these concepts may vary by output format, likely using some combination of boldface, italics, or underlining.

Heads up! We'll talk about customizing presentation later, but it's important to remember that the PreTeXt community separates such "publication" decisions away from the work of "authoring" content.

For users coming from LaTeX, rest assured your mathematical formulas work in PreTeXt. Inline mathmode $ax^2+bx+c=0$ is invoked with `<m>ax^2+bx+c=0</m>`, while display mathematics like

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

is available via `<me>x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}</me>`. (Users from LaTeX will also appreciate that quotes are surrounded with `<q>` in PreTeXt to handle the different way quotation marks are handled in LaTeX vs most other markup languages.)

You may also have lists withinn paragraphs, ordered `<ol>` and unordered `<ul>`, nested as needed. Each list item is represented by `<li>`.

- A single item.
- An item with an ordered list.
    1. First item.
    2. Second item.

Of course, often you have important **blocks** of content to include, such as `<definition>`s or `<claim>`s.

Definition 2.5.1 PreTeXt is an uncomplicated XML language for describing scholarly documents. ◊

Claim 2.5.2 *PreTeXt is the language that will replace LaTeX.*

Proof. Left to the reader. ■

Such content is automatically numbered appropriately. Each of the blocks above is structured with a `<statement>`, and Claim 2.5.2 additionally features a `<proof>`.

Content is often "knowled". A **knowl** is a piece of context-independent information that is useful to transclude elsewhere in the HTML build of your document. For example, in the HTML build for this document, the above proof is knowled by default, and clicking the referenced "Claim" in the previous paragraph expands its knowl to reveal the claim for the reader.

Note 2.5.3 Because this document was edited directly on GitHub using Codespaces, and served with GitHub pages, finding its source is simple: head to its repository[1] and find the corresponding source file[2]. Check the link out to see exactly how each claim, list, etc. in this chapter was marked up!

2.6 Figures and Diagrams

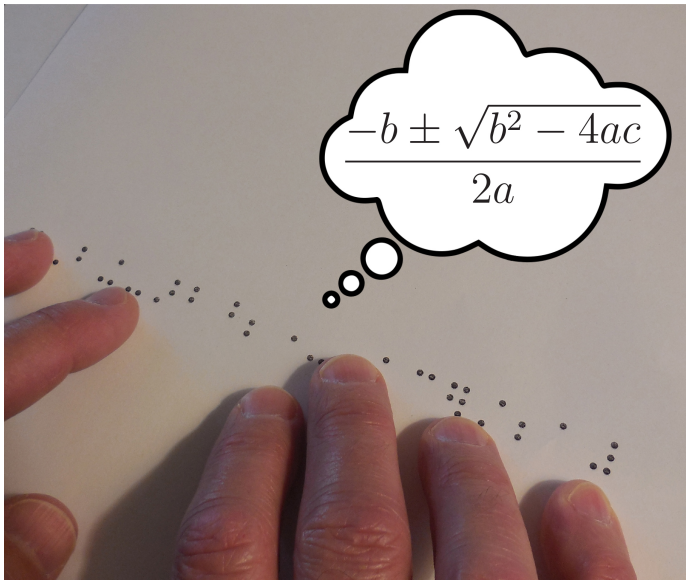

Figure 2.6.1 Photograph taken from AIM Press Release on braille, provided as a JPEG in the project `assets` directory.

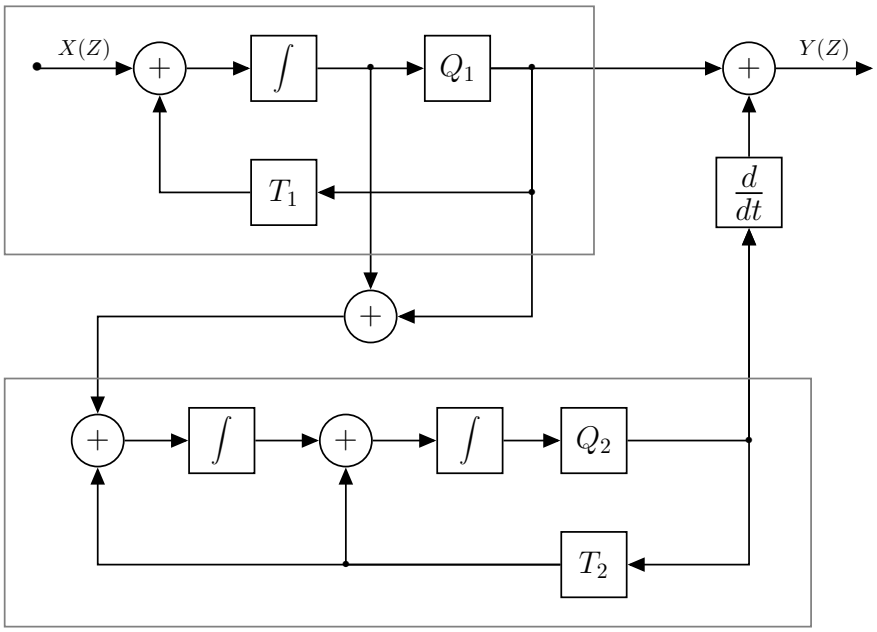

Figure 2.6.2 Electronics Diagram generated with Tikz code

[1] `github.com/StevenClontz/ptx-jmm-2023/`
[2] `github.com/StevenClontz/ptx-jmm-2023/blob/main/source/ch-intro.ptx`

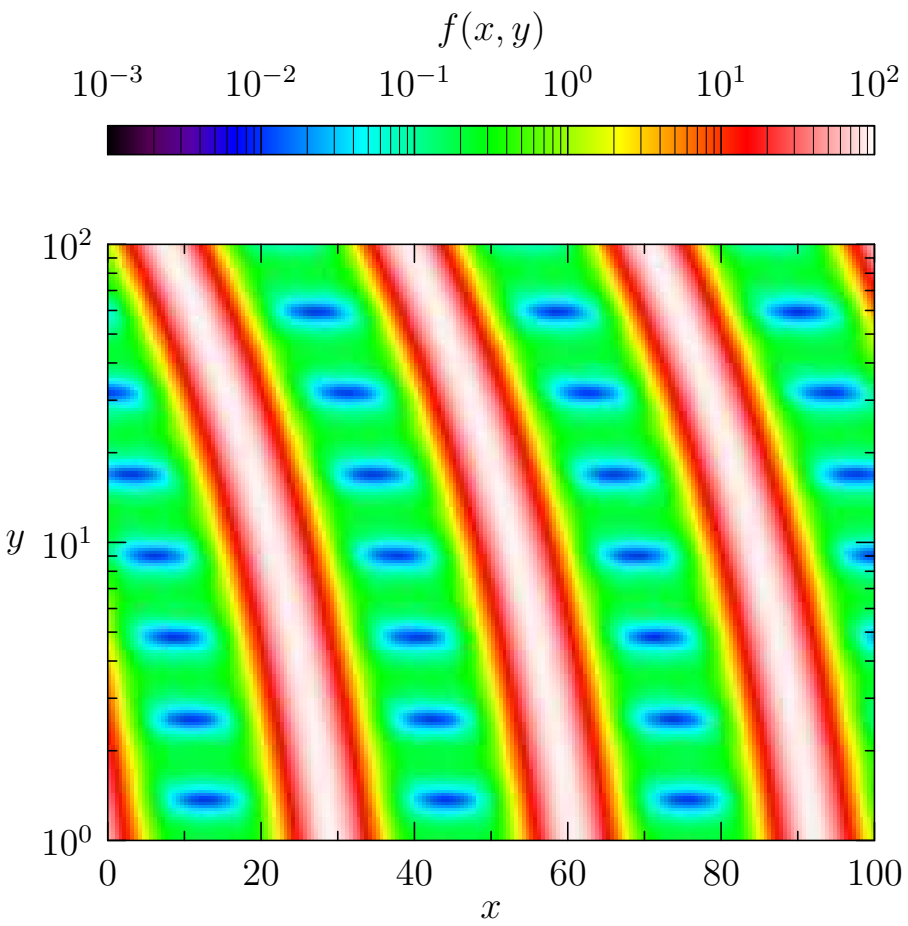

Figure 2.6.3 Contour Plot generated from Asymptote code

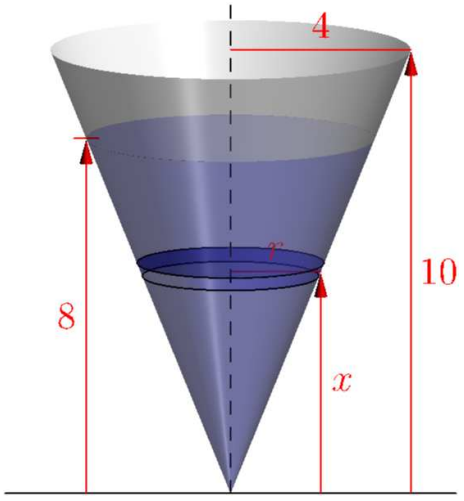

Figure 2.6.4 Work Cone generated from Asymptote code

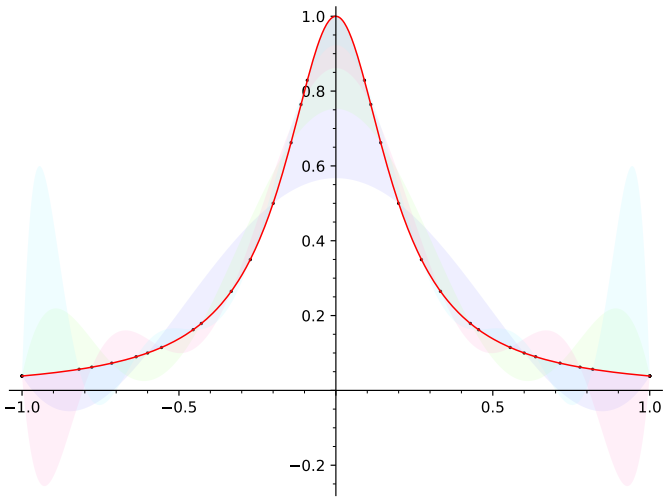

Figure 2.6.5 Polynomial approximations of $f(x) = 1/(1 + 25x^2)$ generated from SageMath code

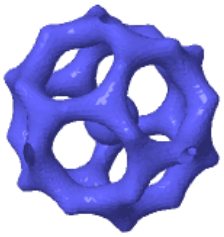

Figure 2.6.6 An implicitly defined 3D surface generated with SageMath code

2.7 Interactives

TODO

2.8 Authoring an Exercise

Checkpoint 2.8.1 Manually-authored exercise. What is $2 + 2$?

Hint. We're being a bit tricky here...

Answer. 22

Solution. 22, where + is the concatenation operator.

Checkpoint 2.8.2 Using WebWork. Compute $1 + 7$.
The sum is __.

Answer. 8

Solution. $1 + 7 = 8$.

Checkpoint 2.8.3 was taken from Linear Algebra for Team-Based Inquiry Learning[1].

Checkpoint 2.8.3 Generated by CheckIt. Consider the following system of linear equations.

$$\begin{array}{rcrcrcrcr} 4\,x_1 & + & 3\,x_2 & + & 18\,x_3 & - & 11\,x_4 & = & -14 \\ -3\,x_1 & + & x_2 & - & 7\,x_3 & + & 5\,x_4 & = & 4 \\ 3\,x_1 & + & 3\,x_2 & + & 15\,x_3 & - & 9\,x_4 & = & -12 \end{array}$$

(a) Explain how to find a simpler system or vector equation that has the same solution set.

Answer.

$$\text{RREF}\left[\begin{array}{cccc|c} 4 & 3 & 18 & -11 & -14 \\ -3 & 1 & -7 & 5 & 4 \\ 3 & 3 & 15 & -9 & -12 \end{array}\right] = \left[\begin{array}{cccc|c} 1 & 0 & 3 & -2 & -2 \\ 0 & 1 & 2 & -1 & -2 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

(b) Explain how to describe this solution set using set notation.

Answer. The solution set is $\left\{ \left[\begin{array}{c} -3\,a + 2\,b - 2 \\ -2\,a + b - 2 \\ a \\ b \end{array}\right] \middle| \, a, b \in \mathbb{R} \right\}$.

Checkpoint 2.8.4 Runestone-powered True/False Question. This statement is true or false.
True or False?

Answer. True.

Solution. True.
Feedback goes here.

Checkpoint 2.8.5 Runestone-powered Matching Problem. A multiple choice question

A. right answer 1

B. right answer 1

C. wrong answer 1

D. wrong answer 2

[1] teambasedinquirylearning.github.io/linear-algebra/

Answer. A, B.

Solution.

A. *Correct.*

answer specific feedback

B. *Correct.*

answer specific feedback

C. *Incorrect.*

answer specific feedback

D. *Incorrect.*

answer specific feedback

Checkpoint 2.8.6 Runestone-powered Parsons Problem. Rearrange the blocks in alphebetical order

- C
- B
- 103
- D
- A

Solution.

- A
- B
- C
- D

Checkpoint 2.8.7 Runestone-powered Matching Problem. Match the letters with their order in the alphabet

A
B
C

Solution.

A	1
B	2
C	3

2.9 Not Just HTML, Not Just PDF

This document is available in HTML format at https://stevenclontz.github.io/ptx-jmm-2023/, and the same document is available in a PDF format at https://stevenclontz.github.io/ptx-jmm-2023/pdf/main.pdf.

Obtaining these formats are as easy as running `pretext build web` and `pretext build print` respectively. PreTeXt supports other types of output as well, including Jupyter notebooks, ePub, and tacticle braille code.

You're encouraged to view authoring in PreTeXt as an *investment*: you may not need the braille output today, but the little extra thought and care required to author in PreTeXt will allow you to provide this version of your document to a blind student tomorrow.

2.10 Wrapping Up

Go check back on the Codespace you created in Section 2.1. If it's up and running, you're ready to move on to the next section!

Chapter 3

Codespace Workflow

Once you create content in PreTeXt, it is time to **build** it, perhaps **generate assets** (if your project has such assets), **view** the result to make sure it looks good, and when you are ready, **deploy** your web output to a live webpage for others to marvel at.

In this chapter we will walk through the steps to do this inside a codespace environment.

3.1 Build and Generate

Inside the VS Code window you opened through codespaces, have a .ptx file open. You can build your entire project in a few different ways.

- Click the green triangle in the top-right corner of the window (hovering will show "Build (select target)"). A window will pop up asking for a **target** to build. To start just select "web".
- In the "Explorer" panel on the left-hand side of the window, you can expand the "PreTeXt Commands" menu and select the Build option.
- You can type `pretext build web` from the terminal (this uses the CLI directly).
- You can use the keyboard shortcut `CTRL+ALT+b` (or perhaps `CMD` instead of `CTRL` if you have a Mac).

If your documentment contains some more complicated elements, you might need to **generate** them for them to show up. The elements that require this are (depending on what your build target is):

- `<latex-image>`
- `<sagemath>`
- `<asymptote>`
- `<youtube>` (for thumbnail previews)
- `<webwork>`
- `<codelense>`

You can generate assets in much the same way you run a build. There is a button on the top-right of the window, and option in the "PreTeXt Commands" menu, you can type `pretext generate` in the terminal, or use the keyboard shortcut `CTRL+ALT+g`.

Note 3.1.1 Note that generating assets requires additional software. If you started the default codespace, then this is not necessarily installed. You can fix this by entering the following command in the terminal:
`sudo bash ./.devcontainer/postCreateCommand.sh`

Alternatively, when you create a codespace, you can click the three dots next to the + and select the devcontainer that has "pretext-full" in its name.

Once you install this software once, you should be good to go as long as your codespace exists.

3.2 Previewing

You can check the output of what you built using the **View** command. Again, you can access this in multiple ways: top-right icon or PreTeXt Commands menu. When you use either of these, you will get a choice for your viewer. We currently suggest using "Live Preview", although on codespace this requires a few extra steps:

1. When the side preview opens, it will ask you if you want to open in an external browser. Click "Open in browser".
2. Now close that new tab that opens, and also close the side panel in VS Code that opened.
3. Finally select "View" again, and it should just work. You can drag the VS Code tab to un-split the window to make it easier to view.

You can also experiment with the CLI view command, by selecting that from the pop-up menu, or by typing `pretext view web` in the terminal. This should pop up a new browser tab with the preview. The only reason we caution against this currently is that the local server that gets started to preview the files will keep running as long as your codespace is active, and we don't understand how this affects resource use.

When you make edits to your source files, you will need to build again, and then refresh the preview window to see the changes.

3.3 Saving your work

Using codespaces will keep all your files "in the cloud" on GitHub's server. As long as you don't delete your codespace, your files will be saved there. However, you will want to **push** these files to your **git repository** on GitHub to make this save perminent. This has the benefit of allowing collaborators to access your files as well (your codespace is unique to your account).

There is a *lot* to learn about git, but luckily using VS Code lets you do everything you need using menus (you don't need to use the command line, unless you want to). Everything can be controlled using the **Source Control** view: it should be third from the top on the very left of the window, an icon with splitting paths, and likely a badge showing how many files you have changed.

Here are the basic concepts you need to understand.

- The program **git** keeps track of all the changes you make to files inside of your **repository** (in this case, the folder containing your project).
- Once you have edited your files and are happy with all of them, you tell git to track the set of changes as a **commit**. This creates a handy *breakpoint* you could return to if you want to go back to an earlier version.

  There are two steps to creating a commit (which you can often do all at once in practice):

  1. You **stage** the files you want to update in the commit.

     You **commit** the stage files including a **commit message**.

  Doing this in two steps can be helpful if you want to commit only some of the files that have changed.
- Once you have one or more commits, you need to sync these changes with GitHub. To "upload" your changes, you **push** the repository. To download changes that you are someone else made, you **pull** the repository.

Now, how do we do these things in VS Code? Start by looking at the Source Control view. You will notice a list of files that were changed. You can click on any of these to see what the changes are (you will see a side-by-side view of the original and updated version).

If you are comfortable staging and committing in one step, you can simply write yourself a short message in the textbox above the big green "Commit" button, and click the button. If you want to stage first, click the + next to each file under "changes" to stage them.

The green button should now turn into a "Sync" button. When you click that, it will do a quick pull and then a push, to sync changes with GitHub.

The only small point about using git is that no all files will be tracked. This is on purpose, since temporary files really should not be "remembered" using this version control setting. Which files or types of files are ignored by git is controled by the ".gitignore" file in your repository.

In particular, we do not track the output of builds. Git is used to track progress on your source, which you build into output at any time. If you want others to be able to see the output of your work without building it themselves, you need to deploy your work.

3.4 Deploy

So you have worked tirelessly to prepare course notes or a book, built and previewed, synced changes using git, and now you are ready to share the results of your efforts with the world. It's time to **deploy** your project.

With our codespace setup this is simple. From the "PreTeXt Commands" menu, click on "Deploy to GitHub". This will automatically take the most recent build of your web target and host it through GitHub Pages[1]. Watch the output pane for a link to your published site. (It can take a few minutes for the site to get set up or updated; there should be another link to view the progress of the GitHub "action" that reports the progress.)

[1] pages.github.com/

Appendix A

Getting Help

Here we collect a number of useful resources to help you when you are stuck. The official PreTeXt site[1] has lots of resources, but we understand it can be overwhelming.

Official documentation. Note that the official PreTeXt Guide[2] can be hard to use because there is so much stuff in it. Additionally, some of the documentation is out of date. Still, if you know where to look, it is a great resource.

Here are some sections that we find especially helpful:

- Basics Reference[3]: A listing of the main elements of PreTeXt including snippets of the code that create them. This is one of the few places in the guide that has examples of the markup.
- Publication File Reference[4]: When you are ready to start changing how your output looks, you can use the **publication file**, which is described in this part of the guide.
- PreTeXt Schema[5]: The official list of elements and where they can go is given in the PreTeXt Schema, which is described here. Also you can check out the schema browser[6] to actually view the schema.
- Getting PreTeXt[7]: If you want to install PreTeXt on your own computer, this early part of the guide gives you directions. It should be updated with information on CodeSpaces soon as well, if you need a refresher.

Finally, note that the search in PreTeXt now works really well, and searching for a feature will usually get you pointed in the right spot.

Examples. The Examples[8] page on the PreTeXt site contains a number of useful live examples. Links are provided to web, pdf, and source (on GitHub). For some of the examples, there is also an *annotated* version available. We find these especially helpful since you can "view source" to see exactly how each bit of the example was marked up in code.

Here are some of the most useful such examples:

[1] pretextbook.org/
[2] pretextbook.org/doc/guide/html/guide-toc.html
[3] pretextbook.org/doc/guide/html/part-basics.html
[4] pretextbook.org/doc/guide/html/publication-file-reference.html
[5] pretextbook.org/doc/guide/html/schema.html
[6] pretextbook.org/doc/schema/
[7] pretextbook.org/doc/guide/html/quickstart-getting-pretext.html
[8] pretextbook.org/examples.html

- Sample Book[9]: This annotated sample book contains a section on interactive exercises[10]. The PreTeXt developers use this book for testing, so you can see the latest (sometimes experimental) features available.
- Sample Article[11]: Not particularly well organized (it is also a proving ground for developers) but this contains almost every variation of every feature of PreTeXt. Using the search and "view source" makes this an invaluable resource.

Community Support. There is a very active google group for support: pretext-support[12]. You should also subscribe to the low-traffic pretext-announce[13] to get updates.

This spring we will host daily virtual "drop-in" sessions to support authoring and devlopement of PreTeXt. Information will be posted to the pretext-announce google group.

[9] pretextbook.org/examples/sample-book/annotated/
[10] pretextbook.org/examples/sample-book/annotated/interactive-exercises.html
[11] pretextbook.org/examples/sample-article/annotated
[12] groups.google.com/g/pretext-support
[13] groups.google.com/g/pretext-announce

Appendix B

Copyright and Licensing

[1] creativecommons.org/licenses/by-sa/4.0

Appendix C

Acknowledgement

We would like to thank the American Institute of Mathematics[1] for sponsoring us to present this Professional Enhancement Program at the 2023 Joint Mathematics Meeting.

[1] aimath.org