Warhorn Classics Documentation

Contents

Preface

This guide is meant to be a repository of information on both *what* we have decided in terms of various style decisions, as well as *how* to accomplish those decisions. Currently the documentation is limited to books, but it is possible that additional information will be added both for other kinds of works.

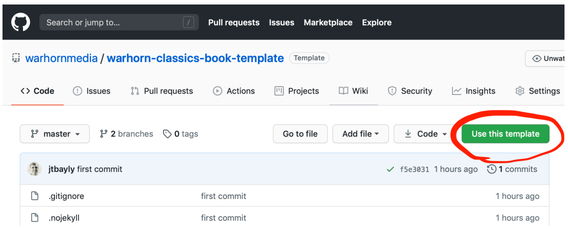
Warhorn Classics uses Bookdown to create all its online versions of books. For technical questions about how to accomplish something that is not covered in this guide, this book will likely answer the question or put you on the path to finding out.

Creating a new book from scratch

0.1 Creating a new repo for a new book on Github

NOTE: This guide assumes you already have the ability to work with Git repositories on Github, including checking out repos, as well as making and pushing commits. (Also, if you are not a power user of Git, we recommend Github's desktop tool. It makes it simple to work with repos without memorizing command-line syntax, and it handles most simple scenarios.)

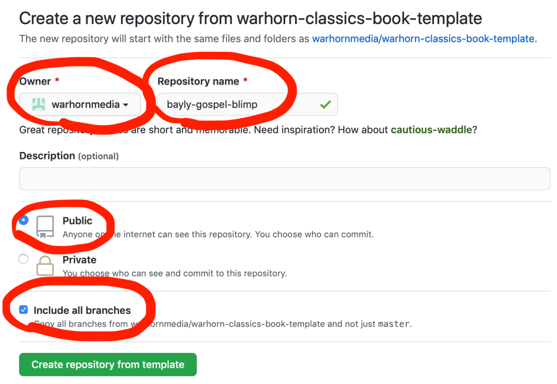
1. Go to our empty book repo on Github and click "Use this template." (You must be logged into Github already.)



jtbayly first commit ✓ f5e3031 1 hours ago 1 commits

.gitignore	first commit	1 hours ago
.nojekyll	first commit	1 hours ago

2. Change the owner to warhornmedia. Enter a repository name using the format "authorlastname-short-book-title". Set the repository to public. And include all branches. Then click "Create repository from template"



Create a new repository from warhorn-classics-book-template

The new repository will start with the same files and folders as warhornmedia/warhorn-classics-book-template.

Owner * warhornmedia

Repository name * bayly-gospel-blimp ✓

Great repos[illegible] are short and memorable. Need inspiration? How about cautious-waddle?

Description (optional)

[✓ Public] Anyone o[illegible]e internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

[✓ Include all branches] Copy all branches from w[illegible]ornmedia/warhorn-classics-book-template and not just `master`.

3. Clone the new repo to your computer and begin making changes

Congrats! You now have a new book that will rebuild automatically any time you push changes to github.

Your next task will be to update all of these files, and then to add the book content. You will find this information in the following sections.

0.1.1 (Optional) Preparing and using a local build environment

Being able to build a book locally can save time so you don't have to wait for the book to be rebuilt by continuous integration. Typically a local build is done in under 10 seconds, and an online build takes 2-3 minutes. This can really add up if there is a problem you are trying to fix through trial and error.

Instructions:

1. Download RStudio for desktop here.
2. Make sure that LiberationSerif is installed. (TODO: When we finalize fonts, this will need to be updated. It's possible it's already out of date.)

Now you can open the file that ends in .Rproj, and you should be able to click "Build Book" in the "Build" tab in the upper right.

A note on classics-template-files: When you first build the book locally, a new folder will be created in the book folder called classics-template-files. These files come from a separate repository by the same name. They allow us to make changes to the style and front matter of every book from one central location. But once you have these files in your local book folder, they will not be updated with new changes to the design unless you delete the whole classics-template-files folder. (This folder is excluded from tracking git, so you don't need to worry about deleting it or updating it before committing changes to the book.) Thus, if you have not worked on a book for a while, it makes sense to delete this folder from your local book repo before you next work on it.

0.2 What are all these files?

Below is a list of the files in the book template, along with a description of that file and the contents. If the name of a field inside a file is **bold** below, that means you need to customize that field for a new book.

0.2.1 Files you must modify

01-Basic-instructions.Rmd Contains the content of the first chapter of the template book. **Should be deleted or renamed** and its content replaced with the first chapter of the new book. See

_bookdown.yml contains some settings for bookdown.

Contents:

- **book_filename:** Change this to give book downloads, such as PDFs, appropriate filenames. Don't use spaces. Standard: author's last name followed by short title, like so:
  `book_filename: "Ryle-Duties_of_Parents"`

Don't change anything else in this file, unless you need "chapters" to be called something else. If so, make sure you check all the output formats thoroughly and report back. This is currently untested in our books.

_editorial-notes.Rmd Contains two sections that must be updated throughout the editorial process, and is automatically included in the front matter of the book.

Contents: :

- **Text Status** This section should be updated by moving the "**Current status:** –>" to the front of the appropriate line as the project progresses. :

- **Editorial Notes** Any noteworthy editorial decisions should be added to this section.

cover.jpg **Should be replaced** with an image of the book cover. Alex is working on a simple design that will be able to be used for all of the books. Both JPG and PNG files are supported.

index.Rmd Contains many settings for the book and the code to automatically include the Classics front matter in each book. Anything not mentioned below should be left unchanged.

Contents:

- **title, author, and date** must all be changed and should be self-explanatory
- **description:** Short description of the book. Don't use copyrighted content here. Write your own description. This field is useful for SEO. Also, it is displayed together with the cover image when the book is shared on certain social media.
- **params:**
- **pubinfo:** This appears in the front matter and should be updated with the original publication info.
- **scans:** Enter a link to scanned images of the original public domain work on Archive.org, Google books, or elsewhere.
- build: This can be any number. Change the build number if you ever want to force the book to rebuild but haven't changed anything else. It's used for no other purpose.
- **cover-image:** the name of the image file being used as the cover for the book. Make sure to include the extension, such as cover.jpg, or cover.png.
- **url:** the page where the book is hosted. Should be https://warhornmedia.github.io/your-new-github-project-name/ Replace the last part with your project name. Currently that ending "/" is necessary.
- **output:**
  - **bookdown::gitbook:**
    - **config:**
      - **toc: before:** This content is shown above the TOC on the web version of the book. Having the book title (and author last name if it will fit), is helpful for people to know what they are looking at. Change this on the second line.
      - **edit:** where the edit link on each page points to. Should be the GitHub project URL, followed by "/edit/master/%s"

README.md The description of the repo that will be displayed on the Github project page. **Update this** with some simple info and a link to the readable book.

0.2.2 Other files

_output.yml Contains a number of settings for the various output formats. You generally shouldn't need to make changes to this file. However, you can modify the following line to add or remove links to the various formats if necessary. For example, if the PDF won't build or has problems, simply remove it from the list. Or you can add kfx to the list. Other formats might be possible as well.

- `download: ["pdf", "epub", "mobi"]`

book_project.Rproj This file can be ignored. If you use the Rstudio IDE, this is your project file, and you'll probably want to rename it to the book title just so you don't get confused which book project you are working on.

.github folder contains the Github Actions workflow file that rebuilds the book automatically every time a new commit is pushed to Github. You shouldn't need to mess with this folder. In fact, it is a hidden file, so you normally won't see it at all unless you are working on the command line or Github.

.gitignore Specifies which files are not to be tracked in git. In particular, we only want to track book source, not output files (_book), files related to RStudio, or external files used in the build process (classics-template-files). This file already has the right settings. You shouldn't need to mess with this file. In fact, it is a hidden file, so you normally won't see it at all unless you are working on the command line or Github.

0.3 Adding book content

To add content to the book, simply add .Rmd files. .Rmd files are just text files written in R-markdown. R-markdown is just normal markdown with some extra features supported. The details of the features we support are outlined below. (TODO: We should probably add to this documentation the basic rules about how we want our markdown formatted.)

Note: The filename must end with .Rmd, not .md, if you want it included in the book. That is why the README.md file is not build into the book.

All the markdown files found in the main directory will be combined into a book, ordered by filename. So use numbers at the beginning of the files to indicate the order they should go in. Note that index.rmd will always come first, though, and will automatically include the Warhorn Classics front matter.

To *exclude* a file from the book, simply preface the filename with an underscore. For example, "_excludedfile.Rmd" will not be added to the book by default, even if it is in the main directory with all the other Rmd files.

Here is an example list of files in the order they will end up in the book.

```
index.Rmd
00-preface.Rmd
01-The-Sacraments-In-General.Rmd
02-The-Sacrament-of-Baptism.Rmd
03-The-Sacrament-of-the-Lords-Supper.Rmd
```

Note that the numbers at the beginning of each filename are *only* used to get the files into the proper order in the book. It is handy if they can correspond to the chapter number just to prevent confusion, but it is not necessary.

The online version of the book will be split up into separate pages, not just one long web page. The split will happen at each new .rmd file.[1] Thus, in most cases a new .rmd file should be created for each chapter.

NOTE: Each .rmd file must begin with the title of the chapter (or section) it contains. (The proper syntax is described in the next section.)[2]

0.4 Chapters, sections, and more

The primary structure in Warhorn Classics books is determined by various levels of headers, and it can easily be seen in the automatically generated table of contents.

By default, the top level of structure is called a "chapter." However, this word can be changed in _bookdown.yml if necessary for sermons or other types of works where "chapter" is not appropriate.[3]

The start of a chapter is specified in the text with a # followed by the title of the chapter. Here is an example of the proper syntax:

```
# A Long-expected chapter title
```

The chapter title will be given an <h1> tag in HTML, with similar results in other formats.

[1]This can be changed, if necessary, to split by chapter or section, or even turned off completely.

[2]Supposedly the first file must begin with either a chapter ("# Chapter name") *or* a section ("## Section title"). However, it appears that it must be a chapter, as starting with a section causes an error. Latter files may start with a section, instead, if necessary, to split the book into appropriate-length webpages.

[3]The bookdown documentation says it is also possible to split a book into "parts" made up of multiple chapters, as well as add a special "part" called "appendix." However, ebooks and Word documents do not get any "part" information, so we do not support this option in our books.

Chapters can be broken down further into sections, sub-sections, etc. up to 6 levels total, using additional levels of headers and titles:

```
## This is a section. (It will receive an <h2> tag in HTML.)

### Here is a sub-section.

#### And now a sub-sub-section

###### This is the deepest level supported, because <h6> is the deepe
```

0.4.1 Numbering (or not)

Chapters, sections and subsections will all be automatically numbered, unless you exclude them from numbering by adding "{-}" to the end of the line. For example, generally an introduction should not be numbered:

```
# Introduction {-}

Text of the introduction goes here...
```

Or perhaps the subsections in the book are not numbered:

```
# How to write a book

## Getting started

### Arranging your pencils {-}
```

It is possible to assign an ID *and* specify that a section is not to be numbered like so:

```
## Section forty-five {- #sec-45}
```

0.4.2 IDs

Chapters and sections always have an ID so they can be linked to. If one is not set explicitly, one will automatically be given (implicitly) based on the title. This will be discussed later in the references section.

Formatting

0.5 Overview

Formatting should not be used for structural elements such as headers or captions. Our templates will style those elements according to a standard design. Formatting should only be done where the formatting is essential to the text. For example, bold or italics that the author is using for emphasis should be included, whereas if a (sub)section heading is italicized in the source book, that is a question of design.

Formatting is accomplished by using R-markdown. Basically, you can use anything that Pandoc supports. Where Pandoc supports multiple options, we have generally chosen a specific method. If you need to use something not mentioned below, please suggest it be added here.

0.6 Italics

Italicized text is indicated by surrounding it with single asterisks.

```
Here are *a couple* in italics.
```

0.7 Bold

Bold text is indicated by surrounding it with double asterisks.

```
Here are **a few words** in bold.
```

0.8 Small caps

I cannot think of any other circumstance where smallcaps should be used except the word Lord.

```
The [Lord]{.smallcaps} said to my Lord, "Sit at my right hand..."
```

Output: The Lord said to my Lord, "Sit at my right hand…"

0.9 Centering text

NOTE: This still needs to be implemented in the Mobi, and ePub versions. It works in PDF and HTML versions.

I cannot think of a place where text would need to be centered in the text of a book.

```
::: {.center data-latex=""}
Republished by Warhorn Classics
:::
```

Output:

Republished by Warhorn Classics

Special characters

There are a number of special characters that are created using a backslash (\) before another character. These two-character codes are easily visible in the markdown because of the the backslash, where otherwise they would be difficult to notice.

0.10 Non-breaking spaces

Non-breaking spaces prevent two words from being split onto separate lines. There are a variety of cases in which they are necessary, including Scripture references. They are formed by putting a backslash prior to a regular space.

```
1\ Peter\ 1:3
```

Output: 1 Peter 1:3

0.11 Line breaks

Sometimes, you may need to specify that text should start on a new line but remain part of the same paragraph. This can be accomplished by putting a single backslash at the end of a line.

```
This paragraph will continue \
on the next line.
```

Output: This paragraph will continue
on the next line.

0.12 Backslashes

Because backslashes are are special characters, if one needs to appear for some reason in the actual text of a book, it must be 'escaped' using another backslash. In other words, put two backslashes like so in the markdown followed by the output:

Markdown:

```
Here is a backslash \\ in the middle of a sentence.
```

Output: Here is a backslash \ in the middle of a sentence.

Comments

If you ever need to leave a comment in the source but you don't want it to appear in the built books, you can use the following format. It is the standard HTML comment, but with an extra dash to open it. These comments will not even show up in the HTML source. If you use a regular HTML comment with just two dashes, they will show up in the HTML source.

```
<!---
your comment goes here
and here
-->
```

Poetry

**Note: This still needs to be nicely formatted.

Poetry should be designated as such so it can be styled differently from the rest of the text. Please use a fenced div, with the .poetry class, and specify the attribute data-latex=“ “. Here is an example.

```
::: {.poetry data-latex=""}
| Oh when a mother meets on high
| The babe she lost in infancy,
| Hath she not then for pains and fears,
|     The day of woe, the watchful night,
| For all her sorrows, all her tears,
|     An over-payment of delight?
:::
```

Output:

> Oh when a mother meets on high
> The babe she lost in infancy,
> Hath she not then for pains and fears,
>   The day of woe, the watchful night,
> For all her sorrows, all her tears,
>   An over-payment of delight?

Precede each line of poetry with a | and a space. (This makes a line block so line breaks are respected. We used to use a backslash (\\) at the end of each line to do the same.)

If a line of poetry should appear indented further than the other lines in its final form, precede the text of the indented line with spaces. In line blocks, spaces at the start of a line are respected and preserved as simple spaces, and we don’t need to force them by using non-breaking spaces.

Links

0.13 External links

Links are easily added using standard markdown like so:

```
This text needs a link [here](http://google.com).
```

Linking to any place within the book itself is called a reference and is addressed next.

0.14 References (ie internal links)

It is possible to link to content within the book itself without a full URL. For example, if the author refers to another chapter or section of his book, we can add a link to it without needing an actual url. All that is necessary is that we have the ID of whatever we want to reference. Keep reading for details on how to get or assign ids.

Reference syntax:

Simply bracket the words you want to link, followed by a set of parentheses with a pound sign and the id you want to link to. Here is an example:

```
Here is how I would add a link to the section later in this guide whi
```

Output: Here is how I would add a link to the section later in this guide which is about non-english content since its id is "foreign-languages".

0.14.1 Assigning implicit or explicit IDs

Chapters, sections, and sub-sections are automatically assigned ids using their full (lowercase) text, with spaces replaced by dashes and punctuation dropped. For example, this section would be automatically assigned an id of "assigning-implicit-or-explicit-ids". That is an *implicit id*.

If you're not sure what exactly an implicit id will be, and you need to use it, you can build the book and then hover over the chapter or section and the link will include an anchor name which is the id. You can then use it in the text.

However, it is easier to simply assign a manual explicit ID to the section or chapter you want to refer to. In fact, because these automatic implicit ids change with the text, can be hard to know exactly what they will be, and because they can get long, it is generally better to manually assign explicit ids to chapters, sections, and sub-sections if you need to link to them.

Here is an example of assigning an explicit id for a chapter:

```
# My very long chapter title that I don't want to have to type often
```

Images and links are not given implicit ids, but can be assigned explicit ids if necessary. As we will see, endnotes make extensive use of links with explicit ids.

This is the syntax for assigning an id to a link:

```
Here is a [link](https://www.google.com){#my-google-link-id} (to Goog
```

Here is a link (to Google) with its own id.

Internal links use the same syntax. For example:

```
Here is [a link](#comments){#another-link-id} with its own id to the
```

Linking to arbitrary locations

You can even assign an explicit id to an arbitrary location in the text, for the purposes of linking to any particular spot in the text, even if there is no section heading nearby, for example. But we still need something that can be assigned an id. If there is no convenient image or heading to receive an id, a workaround is to create an empty link and assign it an explicit id.

At the end of this sentence is a link without any text or destination that creates a named ID. You will not see it, but it will exist in the HTML, allowing us to link to it below. The markdown to create it looks like this:

```
A sentence ending with a named empty link. [](){#namedEmptyLink}
```

The markdown to reference (link to) that named empty link looks like so:

```
[Here](#namedEmptyLink) is a reference to that named empty link.
```

Here's what it looks like when it is output:

Here is a reference to that named empty link.

0.15 Footnotes

Footnotes are a special kind of link made up of two parts—the link to the footnote and the content of the footnote. The links are *named* in the markdown source text, but when processed, those names aren't seen. Instead, they are auto-numbered consecutively throughout a chapter. Then, the content of each footnote is also numbered and properly placed at the bottom of the page. Footnotes also automatically have a backlink so you can return to where you were reading when you clicked the footnote.

Footnote *links* can be added to a book anywhere, including section headings or chapter titles (but see warning concerning multi-paragraph footnotes below).

The *content* of that footnote is put in its own paragraph, indicated with special sytanx (see below). In the markdown source text, please put the *content* of the footnotes directly under the paragraph where the footnote occurs. This will make it much easier to find and edit the footnote in the source document. Remember, after processing it will show up at the bottom of the page, as expected. To prevent confusion (and because of potential bugs), every footnote in the book *must* have a unique name.

Footnote naming should follow this pattern: chapter number, period, footnote number. So the first footnote in the first chapter would be named "01.01". The third footnote in the fifth chapter would be "05.03". A footnote in the intro could be named "intro.01".

If the structure of a particular book demands a more detailed naming of footnotes, it may be appropriate to lengthen footnote names for clarity's sake. For example, a book with lengthy sections within chapters might have footnotes named as follows: "01.01.01", "01.01.02", etc., throughout chapter 1, section 1; then "01.02.01", "01.02.02", etc., throughout chapter 1, section 2. The most important thing is that each footnote have a unique name.

Remember, we use this pattern of consecutive numbers for the names in order to simplify the process of editing as well as to guarantee unique names. However, these names are not used in the final book. Automatic consecutive numbers are. So if you accidentally skip a number, the markdown source text numbers will *not* match the numbers in the processed book from that point to the end of the chapter. (Automatic footnote numbering restarts at 1 at the beginning of each chapter.)

Simple footnote syntax:

```
Footnotes are often found at the end of a sentence like so.[^01.01]

[^01.01]: Here is the footnote's content, in its own paragraph with a

Now the main text of the book continues.
```

Multi-paragraph footnote:

Indent latter paragraph(s) in a footnote by four spaces to indicate that the footnote is continuing.[4]

```
Longer footnotes can also appear in books.[^01.02]

[^01.02]: Another (longer) footnote. It is possible to have multiple

    Simply put four spaces before the next paragraph(s) to indicate t

Here is the next paragraph of the book. It is no longer part of the f
```

Warning: Multi-paragraph footnote links *cannot* be put inside bold or italic text or in chapter or section headers, because of a bug in Pandoc. However, they do work within blockquotes. There may be other similar places yet to be found. Check for footnotes inside formatted text if rendering the PDF fails with an error something like this: `! Paragraph ended before \text@command was complete.` If you find another place where multi-paragraph footnotes don't work, please update this documentation.

0.16 Endnotes

Endnotes are like footnotes that appear at the end of the book instead of the bottom of the page. However, Bookdown only supports one type of "note" by default. All of the notes in a book will be either treated as footnotes or endnotes, depending on the settings. Since our settings instruct Bookdown to treat notes as footnotes, the process of adding endnotes is a bit more manual. In fact, endnotes are not special links at all. We are just making use of Bookdown's references (ie internal links) feature to standardize a process of creating bi-directional links to and from the end of the book where the endnotes are.

The first step in the process is to create a separate endnotes.Rmd file[5]. This will

[4]Here is my multiparagraph footnote, just to prove that it works.

See? Second paragraph works just fine. :)

[5]There is an endnote file in this style guide, for demonstrations' sake. And this is a demonstration of an endnote within a footnote. 2

result in a separate chapter at the end of the book for the content of the footnotes. Then, two separate named links need to be created that point to each other. The first is in the text of the book where the endnote reference is. The second is in the endnote file at the beginning of the endnote. This creates a link to the endnote from the body of the main text, and as well as a backlink, so you can return to where you were reading. The end of this sentence has an endnote to demonstrate what an endnote normally looks like. (1)

NOTE: Since endnotes are simply internal links, they are not able to be styled differently (such as being in superscript) by default. If need be, there may be ways around this limitation. If a text contains both footnotes and endnotes, we will likely need to figure out how to distinguish between footnotes and endnotes a bit more clearly in the main body of the text, since both will simply be numbers.

0.16.1 Creating an endnote

To make it so we can link to endnotes, we need to give each one an ID. The only book with endnotes so far is Buchanan's *Justification*. Each endnote has a title in that book, such as "Note 2, p. 21", so we used the lowest level of header for the endnote title, backlinked it, and gave it an explicit ID like so:

```
###### [NOTE 2, p. 21](#en.p1.02.backlink) {- #en.p1.02}

endnote text goes here...
```

The endnote ID should be named as follows:

en.chapternumabbrev.endnotenumber

So the above endnote is the second endnote in opening of Part 1 before you get to chapter one. Here's how you'd name the ID for the third endnote in chapter 1:

#en.ch01.03

0.16.2 Linking to an endnote

To link to the endnote from within the text, simply follow the instructions for references (internal links) above, and use an explicit id so it can be backlinked. Note that the linked text is not automatic in any way, and can be set to any text. For example, you could make the link "Footnote 1", though that might be too disruptive to the flow of the text, and it is not likely that the original book had that. Still, it is important to understand that the "2" in the example below is not meaningful to bookdown, except as the text to be linked. Here is an example of what it looks like to link to an endnote in the markdown, along with providing a named reference so a backlink can be created:

```
This sentence ends with a link to an endnote in parentheses.([2](#en.
```

Foreign languages

Any text in a foreign language, regardless of how long or short, should be indicated as such with the appropriate two-letter language code, which can be found here. Here is the what the markup looks like for a couple of the most common languages you might need.

Latin:

```
In Latin, the word "[sacramentum]{lang=la}" means "mystery."
```

Output: In Latin, the word “sacramentum” means “mystery.”

Greek:

```
In English, we have transliterated the Greek word "[     ]{lang=el}"
```

Output: In English, we have transliterated the Greek word “μυστηριον” as “mystery.”

NOTE: Do not mark any words as English (ie. “lang=en”). You might be tempted to when, for example, there are a few words in english in a larger Latin section. The default language of the book is English, and attempting to mark any specific words as English will now cause errors when building the PDF.

Typos in the original

Typos can and do appear in the original source documents at times. Our policy is to correct obvious typos, such as spelling errors, making a note of them in the text like so.

```
<!-- Original typo: "profssing" -->
```

Stylistic changes, such as switching from British to American quotes are not considered typos and do not need to be noted.

However, sometimes a word may be missing, or the wrong word inserted. If you believe you have found a typo, but correcting it changes the meaning of the text, please submit it to the editorial team for evaluation. If a change is made in such a circumstance, it will be noted visibly in the text, either through a footnote or an editorial bracket.

Images

Images are likely to be rare, but when used, Pandoc's built-in functionality is not enough. Images are to be placed in a sub-folder called "images" and included using knitr commands:

Centered and 50% width image:

~~~
```{r, echo=FALSE, fig.align='center', out.width='50%'}
knitr::include_graphics("images/sepialogo.png")
```
~~~

Here is what it looks like when used:

WARHORN
CLASSICS

Troubleshooting

What to do if your build is failing:

1. Especially if you just added the OCR'd text of the book, do a search for backslashes in the text. They are a special character and will cause problems unless they are used for the specific things mentioned above.
2. Make sure every .Rmd file starts with either # or ##.
3. If you are working on footnotes, make sure you didn't try to insert a footnote reference inside bold or italic text or from within chapter or section headers *if* the footnote itself has multiple paragraphs. (See warning in footnote section.)
4. Take a look at the log in the Github Action tab, and see if you can find where it failed. (Look for a red line.) Sometimes it will tell you what went wrong, but often the best hint is a number of lines earlier, so scroll up a bit looking for a root cause.

Editorial Policies

Convert British spelling? Convert British or antiquated quotation styles?

Technical details of our automated build system

0.17 Github Actions

We use Github Actions to build books automatically every time a new commit is pushed to a repo. We have our own custom actions repo, that can be used together or separately to build books. The workflow that runs is included with the book's files at `(book)->.github->workflows->deploy_bookdown.yml` It simply calls two actions

1. render-classics-work (see description below)

2. github-pages-deploy-action

   Deploys the final built book to gh-pages branch of the same book's repo. Each repo has Github Pages configured to host a public website of the static files found in its gh-pages branch. So this is the step that takes the newly-built book and updates the publicly accessible website and downloadable files.

0.17.1 render-classics-work

This is a composite action that calls other actions to completely automate building a Warhorn Classics work. In order it does the following:

1. calls our setup-bookdown custom action (see below), which sets up the virtual machine with the necessary software
2. downloads the Warhorn Classics template files (see below) so they are available for the build process
3. runs the classics_extras.sh (see below) script which currently just installs whatever fonts we want to be available
4. calls our build-book custom action (see below) which builds the book

0.17.2 setup-bookdown

This custom action installs a bunch of software and sets up the environment for building the book. Default versions of software are defined (and can be found in the custom-actions readme). Custom versions can also be specified when the action is called.

0.17.3 classics_extras.sh

This script decrypts and installs the fonts included in the classics template files. NOTE: This script is written particularly for MacOS. If we ever want to transition to a Linux VM for our actions, this will need to be modified.

0.17.4 build-book

This custom action runs buildscript.sh (see below), and if the build fails, uploads the book folder as an artifact to help with debugging.

0.17.5 buildscript.sh

This script renders whichever versions of the book are specified in the "download" line of the book's _output.yml file. If necessary for the particular files to be produced, this script will install additional software, such as Calibre or kindle-previewer.

0.17.6 classics-template-files repo

This repo contains a variety of files that are used by our Warhorn Classics books.

- css->warhorn-classics.css is the CSS customization we've done to change the look and feel of the web versions of our books.
- The fonts folder contains some zipped free fonts, and a zipped and encrypted non-free font that we (can) use in the PDF version of our books.
- html->after_body.html is included in every page of the web version of the book.
  - adds refTagger js so Bible verses are auto-linked with popups of the verse
  - adds some js that provides a fix for "split_by: rmd" putting footnotes and arrows in wrong place sometimes
- html->feedback.html *currently unused*
- html->in_header.html Installs our shortcut icon, Adobe fonts, and Google fonts
- images folder has various images used in our designs
- latex->before_body.tex *currently unused*

- latex->preamble.tex changes the PDF design to look the way we want
- rmds->classics-frontmatter.Rmd gets included at the start of every Warhorn Classics book and automatically includes the entire "About this book" page

0.18 Making changes to the build system or templates

Any time a change is made to the custom actions repo or the classics template files, the changes should be verified both to be working properly and that nothing else broke. A good test book is Bannerman's Sacraments. Here is an elementary list of things to check:

Footnotes, endnotes, crossreferences, TOC,

1. Web version
    a. Search working
2. PDF version
    a. colored links
    b. greek words working (including diacritics)
    c. page numbering
3. ePub version

Endnotes

NOTE 1 Here is the text of an endnote.

NOTE 2 Here is the text of an endnote that is referenced from a footnote.