Write Quarto books with Bioconductor

Table of contents

What are `BiocBooks`?

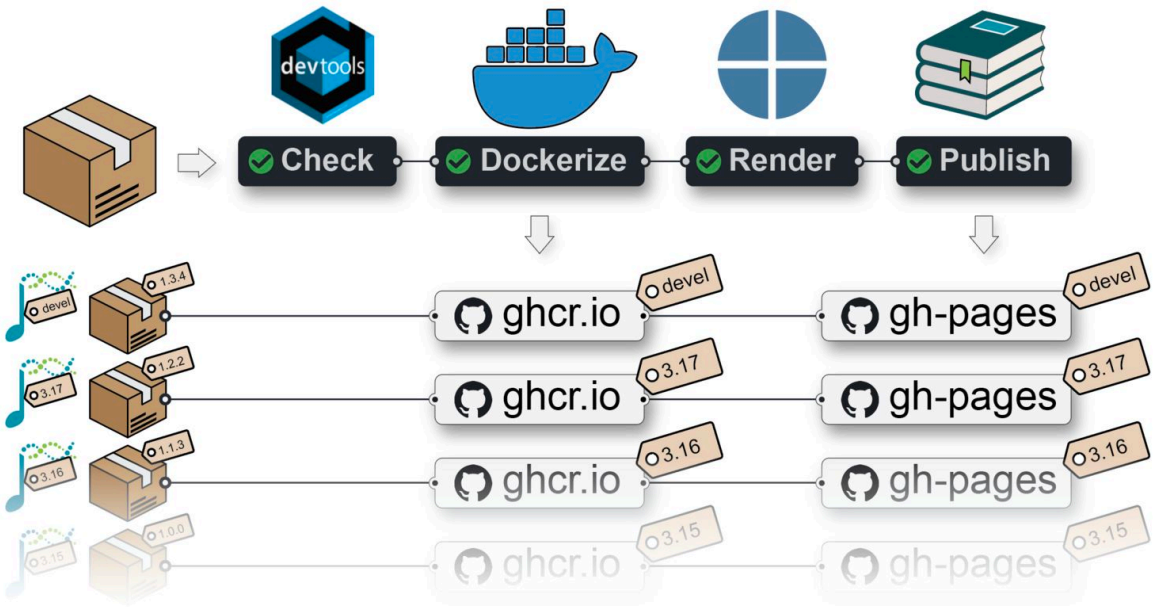
Package: BiocBookDemo **Authors:** Jacques Serizay [aut, cre] **Compiled:** 2023-11-03 **Package version:** 1.1.1 **R version:** R version 4.3.1 (2023-06-16) **BioC version:** 3.18 **License:** MIT + file LICENSE

`BiocBooks` are **package-based, versioned online books** with a **supporting `Docker` image** for each book version.

A `BiocBook` can be created by authors (e.g. `R` developers, but also scientists, teachers, communicators, ...) who wish to:

1. *Write*: compile a **body of biological and/or bioinformatics knowledge**;
2. *Containerize*: provide **Docker images** to reproduce the examples illustrated in the compendium;
3. *Publish*: deploy an **online book** to disseminate the compendium;
4. *Version*: **automatically** generate specific online book versions and Docker images for specific Bioconductor releases.

> Tip
>
> A {`BiocBook`}-based package hosted on **GitHub** with a branch named `RELEASE_X_Y` provides:
>
> - A **Docker image**: hosted on ghcr.io;
> - An **online book** (a.k.a website): hosted on the GitHub repository `gh-pages` branch;
>
> Both are built against the specific Bioconductor release `X.Y`.
> A {`BiocBook`}-based package submitted to **Bioconductor** also lead to the online book being independently built by the **Bioconductor Build System (BBS)** and deployed to `https://bioconductor.org/books/<bioc_version>/<pkg>/`


devtools
Check
Dockerize
Render
Publish
devel
1.3.4
ghcr.io
devel
gh-pages
devel
3.17
1.2.2
ghcr.io
3.17
gh-pages
3.17
3.16
1.1.3
ghcr.io
3.16
gh-pages
3.16
3.15
1.0.0
ghcr.io
3.15
gh-pages
3.15

What is this {BiocBook} package?

The {BiocBook} **package** offers a streamlined approach to creating `BiocBook`s, with several important benefits:

- The author creates a {`BiocBook`}-based package without leaving R;
- The author writes book chapters in `pages/*.qmd` files using enhanced markdown;
- The author can submit its {`BiocBook`}-based package to Bioconductor.

The containerization and publishing of the new {`BiocBook`}-based package is automated:

- A Github Action workflow **generates different Docker images** for different Bioconductor releases, with the packages used in the book pre-installed;
- A Github Action workflow **publishes different book versions** for different Bioconductor releases.

Main features of `BiocBooks`

Fully compatible with the *Bioconductor Build System*

When a {`BiocBook`}-based package is accepted into Bioconductor, it is automatically integrated into the *Bionconductor Build System* (BBS).

This means that it is getting built using `R CMD build --keep-empty-dirs --no-resave-data .`. This triggers the rendering of the book contained in `/inst/`. Book packages built by the BBS are then automatically deployed and are eventually available at `https://bioconductor.org/books/<bioc_version>/<pkg>/`.

Automated versioning of Docker images

A separate Docker image is built for each branch (named `devel` or `RELEASE_X_Y`) of a {`BiocBook`}-based **Github repository**.

Each Docker image provides pre-installed `R` packages:

- Bioconductor release `X.Y`;
- Specific book dependencies from Bioconductor release `X.Y` (listed in `DESCRIPTION`);
- The book package itself

The Docker images also include a `micromamba`-based environment, named `BiocBook`, in which all the softwares listed in `requirements.yml` are installed.

For example, `Docker` images built from the {`BiocBookDemo`} package repository are available here:

ghcr.io/js2264/biocbookdemo

> Get started now
>
> You can get access to all the packages used in this book in < 1 minute, using this command in a terminal:

Listing 0.1 `bash`

```bash
docker run -it ghcr.io/js2264/biocbookdemo:devel R
```

Automated versioning of the online book

Regardless of whether the book package is submitted to Bioconductor, a Github Actions workflow publishes individual online books for each branch (named `devel` or `RELEASE_X_Y`) of a `BiocBook`-based **Github repository**.

For example, the online book version matching the `devel` version of the {`BiocBook`} package is available from:

http://js2264.github.io/BiocBookDemo/devel/

RStudio Server

An RStudio Server instance based on a specific Bioconductor `<version>` (`devel` or `RELEASE_X_Y`) can be initiated from the corresponding `Docker` image as follows:

Listing 0.2 `bash`

```bash
docker run \
    --volume <local_folder>:<destination_folder> \
    -e PASSWORD=OHCA \
    -p 8787:8787 \
    ghcr.io/<github_user>/<biocbook_repo>:<version>
```

The initiated RStudio Server instance will be available at https://localhost:8787.

Further instructions regarding Bioconductor-based Docker images are available here.

Acknowledgments

This works was inspired by and closely follows the strategy used in coordination by the Bioconductor core team and Aaron Lun to submit book-containing packages (from the `OSCA` series as well as `SingleR` and `csaw` books).

- @OSCA
- @SingleR
- @csaw

This package was also inspired by the `*down` package series, including:

- @knitr
- @bookdown
- @pkgdown

Session info

Click to expand

References

Preamble

This page is kept empty on purpose.

1 `BiocBook` versioning system

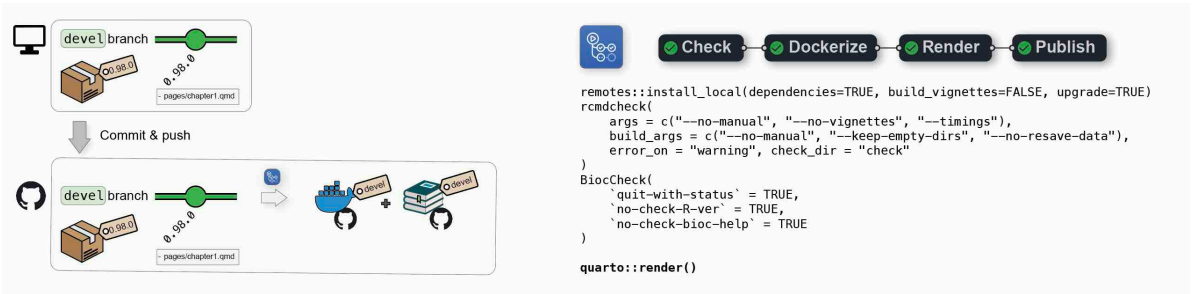
> ⚠ Important points
>
> - Any **package** built using the {`BiocBook`} package is itself a {`BiocBook`}-based package;
> - As such, it follows the release schedule from `Bioconducor`;
> - The `pages/` folder contains a number of pages which are rendered as a **website** using Quarto;
> - The **name** of the branch (`devel` or `RELEASE_X_Y`) is ***crucial***, as it is used to select a Bioconductor version to 1) build a **Docker image** for the {`BiocBook`}-based package and 2) render the `BiocBook` **website**.

1.1 Continuous Integration and Continuous Delivery

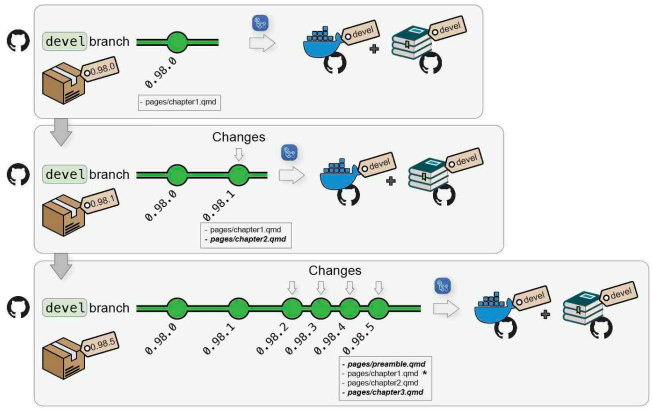
1.1.1 From local to Github

When pushing a {`BiocBook`}-based **package** from a local `devel` branch to Github, (e.g. when writing new articles), two jobs are automatically triggered from a single Github Actions workflow:

1. First, a **Docker image** will be created to build the {`BiocBook`} package against the Bioconductor `devel` branch and push the resulting image to `ghcr.io/<github_user>/<package_name>:devel`
2. Then, this image will be used to render the `BiocBook` **website** and deploy it to `https://<github_user>.github.io/<package_name>/devel/`



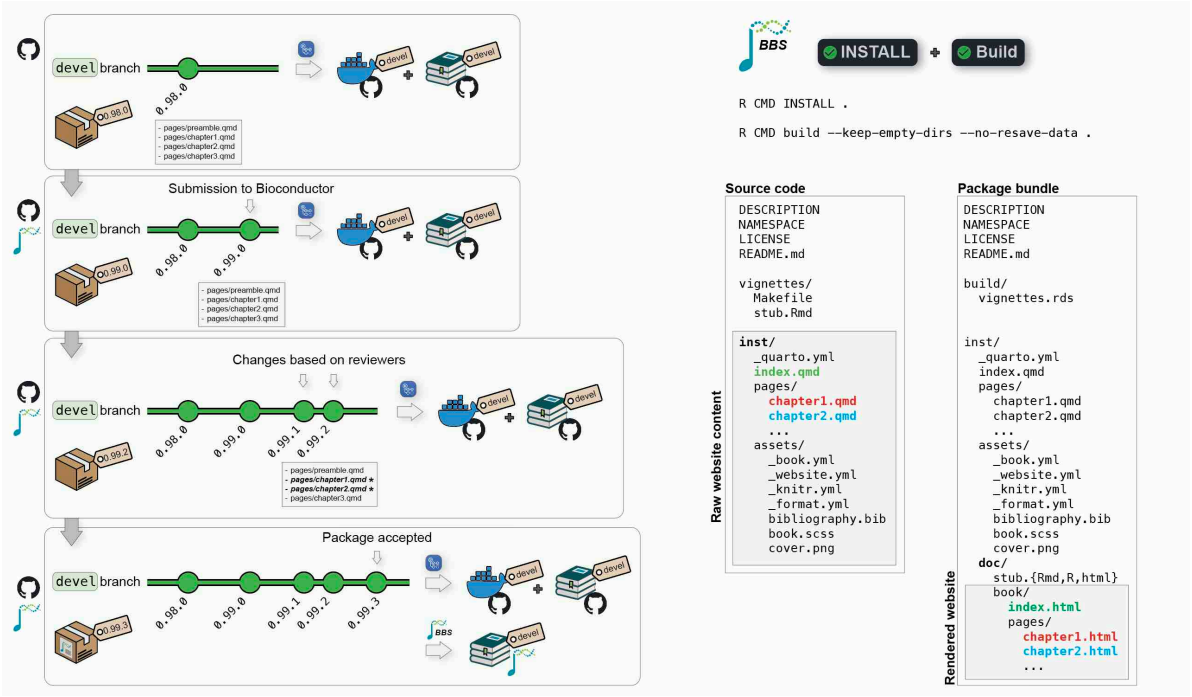
Additional commits on the `devel` branch will trigger regeneration of the `devel` Docker image and the online book `devel` version.



1.1.2 Package submission to Bioconductor

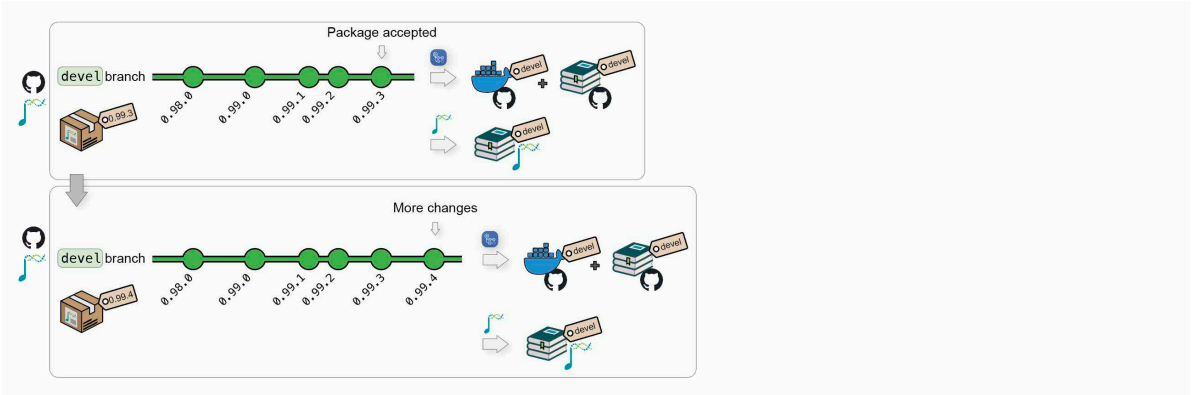
Submission to Biconductor can follow the same reviewing process as other standard packages.

1. The author submits a book package to `Bioconductor/Contributions`;
2. Once review starts, the package gets tested by Bioconductor's *Single Package Builder* (*SPB*);
3. The author pushes changes to its Github repository; this will update the `devel` Docker image and the online book `devel` version;
4. Regularly, the author bumps versions and push to Bioconductor's `upstream` branch to trigger a new *SPB* test run;
5. Once the *SPB* returns no error/warnings, the package may be accepted;
6. When a book package is accepted, it starts being regularly built by the *Bioconductor Build System* (*BBS*).
7. The *BBS* build automatically renders the book and deploys it online.



At all time, the author's local `devel` branch should be synchronized with its Github remote `origin` as well as the Bioconductor `upstream` remote.

- Any commit pushed to the remote Github `origin` remote will trigger a new Github Actions workflow and regenerate the `devel` Docker image and the online book **served by Github**.
- Any commit pushed to the remote Bioconductor `upstream` remote will result in a new build by the **BBS** and an update of the **Bioconductor's hosted book**.

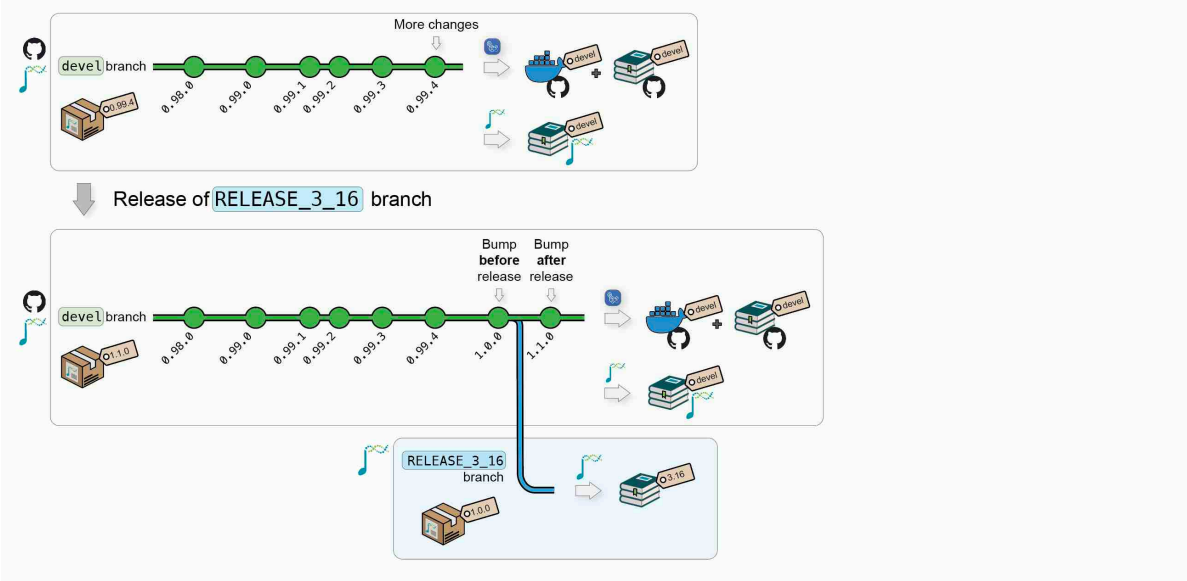


1.1.3 New Bioconductor releases

When Bioconductor releases a version `X.Y`, the core team will automatically create a new `upstream:<package_name>@RELEASE_X_Y`. This will automatically trigger new builds of the **Bioconductor's hosted book** against the new release.

When this occurs, the `upstream` branch can also be pulled to `origin:<package_name>@RELEASE_X_Y` to:

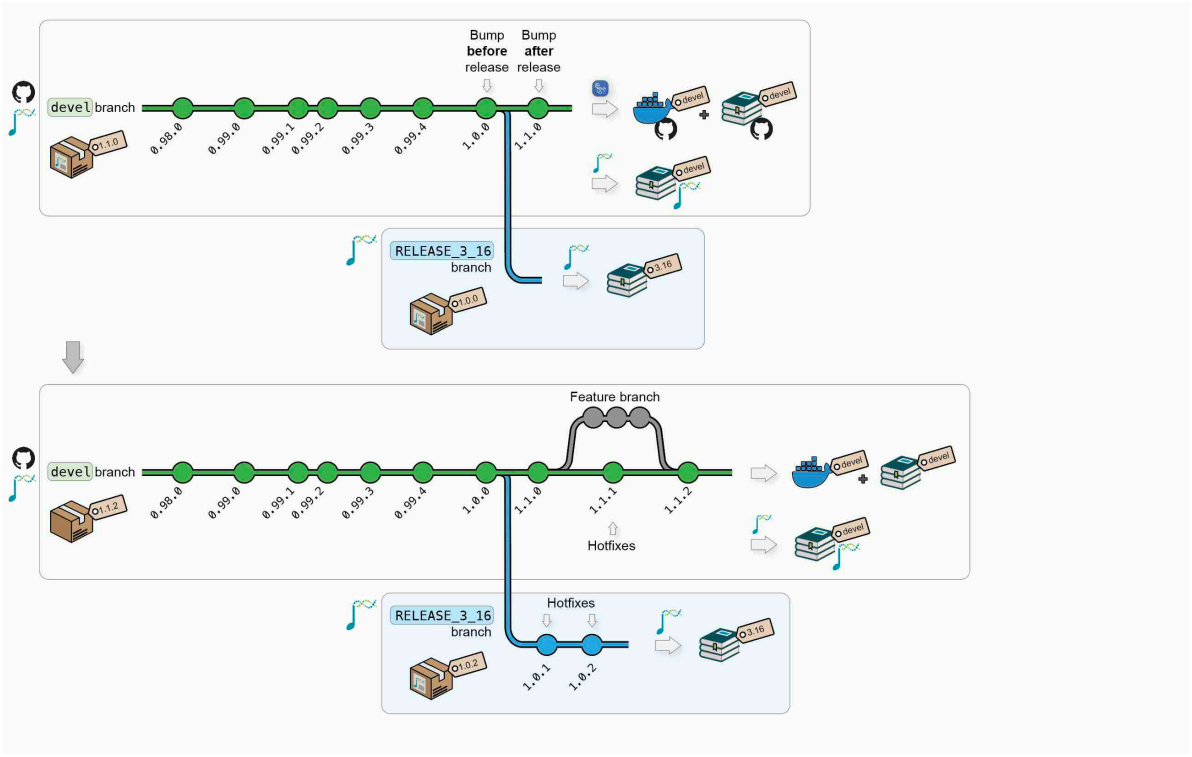
1. Create a **Docker image** @ `ghcr.io/<github_user>/<package_name>:RELEASE_X_Y`, with the book **package** installed using Bioconductor `X.Y`;
2. Publish the book **website** to `https://<github_user>.github.io/<package_name>/X.Y/`, using packages from Bioconductor `X.Y`.



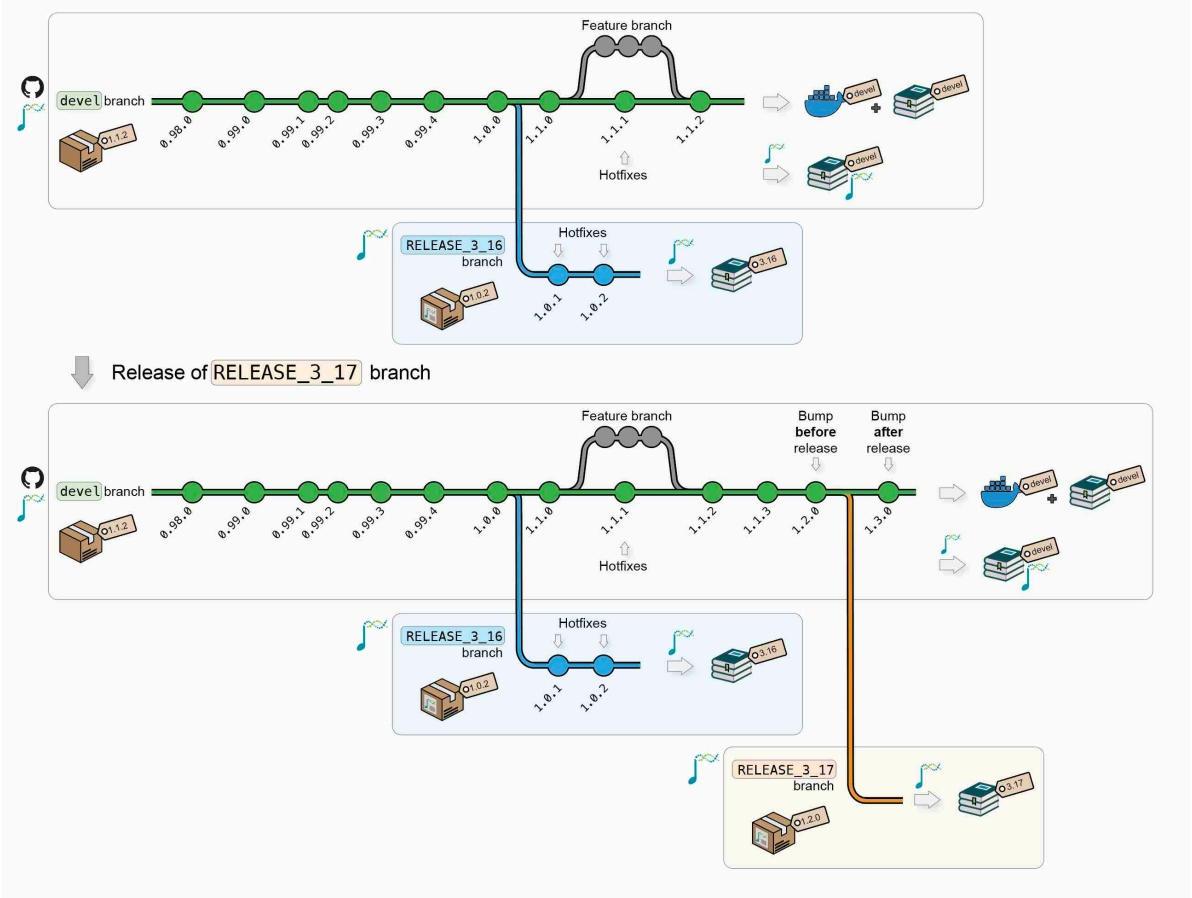
> **Note**
>
> A {`BiocBook`}-based **package** can follow its own release life cycle if the autor does not intend to submit it to Bioconductor.
> If the author of a {`BiocBook`}-based **package** intends to submit this package/book/website to Bioconductor, the Bioconductor submission and release life cycle:
>
> - When developing a {`BiocBook`}-based **package** (at the submission and during review), the **package** version should be between `0.99.0` and `1.0.0`.
> - Once the submission is accepted and Bioconductor releases a new version `X.Y`, the {`BiocBook`}-based **package** version will automatically be bumped to `1.0.0` in Bioconductor release `X.Y` and to `1.1.0` in the continuing Bioconductor `devel`.

1.1.4 Updates

After new releases, updates and/or hot fixes can still made, both on the latest Bioconductor release and on the `devel` branch. New commits will automatically trigger the regeneration of the Docker image and the online book for the modified branch(es).



Additional Bioconductor releases will generate new versions of the Docker image and of the online book.



1.2 Access to versioned Docker and online book

1.2.1 Docker images versioning

The different versions of a `BiocBook` **Docker image** are availabed at the following URL:

`ghcr.io/<github_user>/<package_name>`

For example, for this package ({`BiocBookDemo`}), the following **Docker image** versions are available:

- ghcr.io/js2264/biocbookdemo:devel
- ghcr.io/js2264/biocbookdemo:3.17
- ghcr.io/js2264/biocbookdemo:3.16
- ghcr.io/js2264/biocbookdemo:3.15

1.2.2 Website versioning

The different versions of a `BiocBook` **website** are hosted at the following URL:

`https://<github_user>.github.io/<package_name>/<version>`

For example, for this package ({`BiocBookDemo`}), the following **website** versions are available:

- https://js2264.github.io/BiocBookDemo/devel/
- https://js2264.github.io/BiocBookDemo/3.17/
- https://js2264.github.io/BiocBookDemo/3.16/
- https://js2264.github.io/BiocBookDemo/3.15/

1.3 Does this really work?

The `BIOCONDUCTOR_DOCKER_VERSION` variable is set in all `Bioconductor` Docker images. For instance, this version of the `BiocBookDemo` package online book relies on:

```
Sys.getenv("BIOCONDUCTOR_DOCKER_VERSION")
##  [1] "3.18.25"
```

> **Tip**
>
> Note that this variable will always match the `X.Y` version returned by `BiocVersion` used to render the online book.
>
> ```
> packageVersion("BiocVersion")
> ##  [1] '3.18.0'
> ```

1.4 So what packages can I use?

Any package that has been released in the Bioconductor version you are using (in this book version, this is 3.18.0).

The `BiocBaseUtils` package is available in Bioconductor since `3.16`, while the `BiocHail` package was only made available in `3.17`. `CuratedAtlasQueryR` has recently been accepted in `3.18` (current `devel`, on Fri Nov 3 15:43:09 2023). Let's check this!

```
packageVersion("BiocVersion")
##  [1] '3.18.0'
BiocManager::available("BiocBaseUtils")
##  [1] "BiocBaseUtils"
BiocManager::available("BiocHail")
##  [1] "BiocHail"
BiocManager::available("CuratedAtlasQueryR")
##  [1] "CuratedAtlasQueryR"
```

2 Writing a {BiocBook} package

2.1 Register a Github account in R

> Tip
>
> Skip this section if your Github account is already registered. You can check this by typing:
>
> ```
> gh::gh_whoami()
> ```

2.1.1 Creating a new Github token

```
usethis::create_github_token(
    description = "BiocBook",
    scopes = c("repo", "user:email", "workflow")
)
```

This command will open up a new web browser. On the displayed Github page:

- Select an Expiration date;
- Make sure that at least `repo`, `user > user:email` and `workflow` scopes are selected;
- Click on "Generate token" at the bottom of the page;
- Copy your Github token displayed in the Github web page

2.1.2 Register your new token in R

```
gitcreds::gitcreds_set()
```

Paste your new Github token here and press "Enter".

Saving your Github token for later use

On Linux, `gitcreds` is generally not able to permanently store the provided Github token. For this reason, you may want to also add your Github token to `~/.Renviron` to be able to reuse it. You can edit the `~/.Renviron` by typing `usethis::edit_r_environ()`, and define the `GITHUB_PAT` environment variable:

Listing 2.1 `.Renviron`

```
GITHUB_PAT="<YOUR-TOKEN>"
```

2.1.3 Double check you are logged in

```
gh::gh_whoami()
```

2.2 `BiocBook` workflow

2.2.1 Initiate a {`BiocBook`} package

2.2.1.1 R

Creating a `BiocBook` in `R` is straightforward with the {`BiocBook`} package.

```
if (!require("BiocManager", quietly = TRUE)) install.packages("BiocManager")
if (!require("BiocBook", quietly = TRUE)) BiocManager::install("BiocBook")
library(BiocBook)
biocbook <- init("myBook")
```

The steps performed under the hood by `init()` are detailed in the console. Briefly, the following steps are followed:

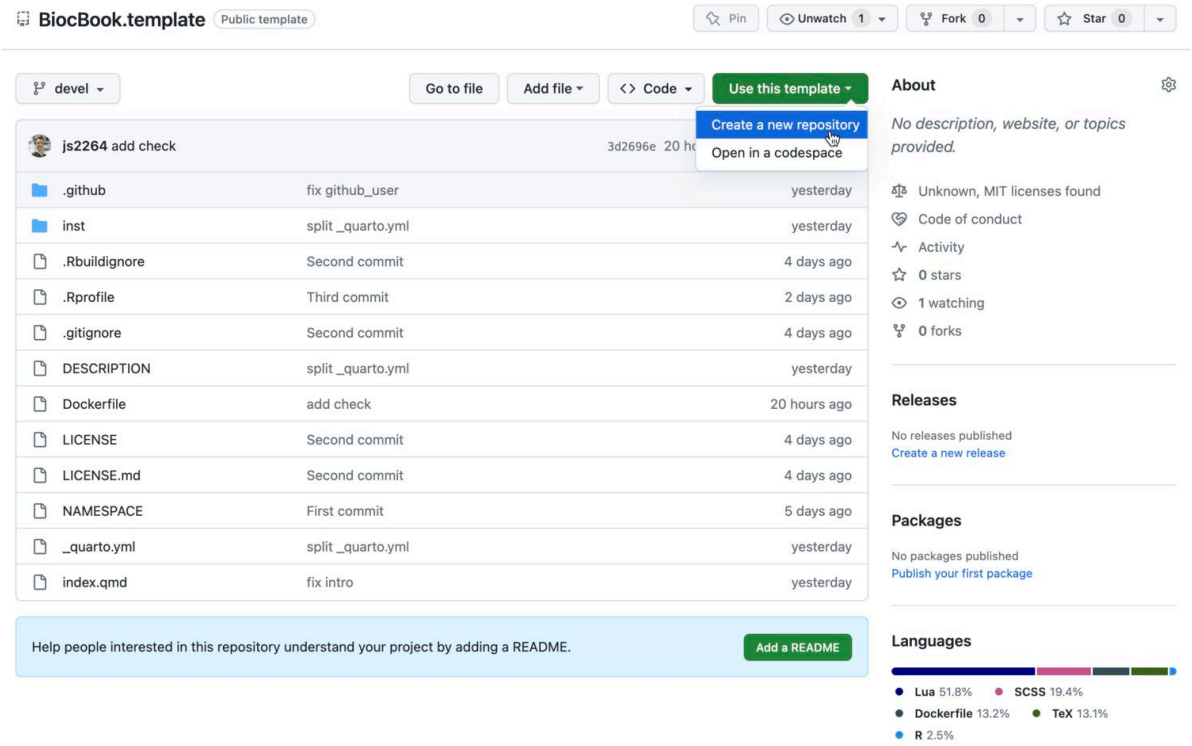
1. Creating a local git repository using the `BiocBook` package template
2. Fillout placeholders from the template
3. Push local commits to your Github account, creating a new GitHub repository

2.2.1.2 VS Code

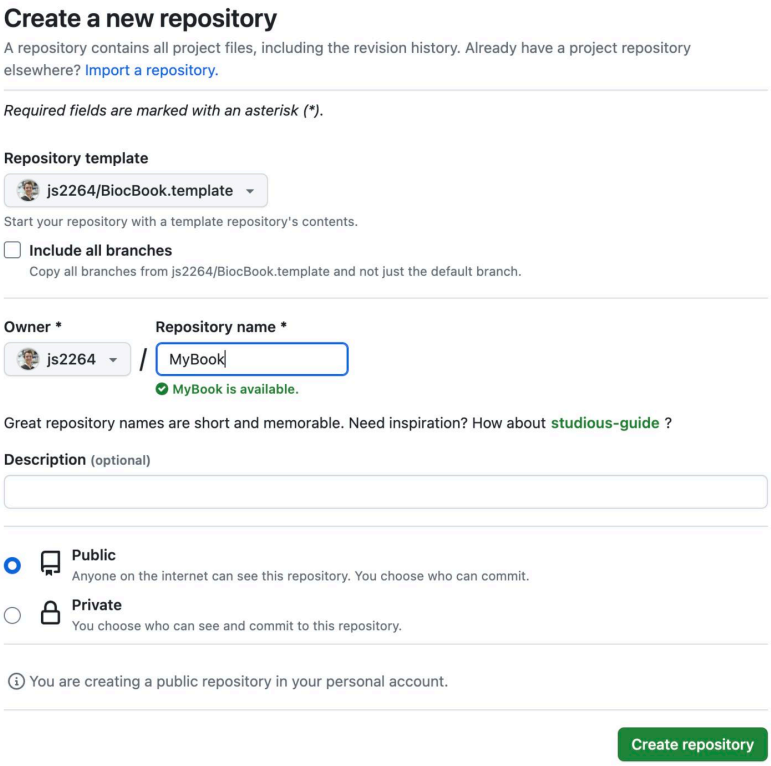
> ⚠ This approach is significantly more hazardous. It is highly recommended to stick to the `init()` helper function from the {`BiocBook`} package.

2.2.1.2.1 * Use the {`BiocBook.template`} package template

This template can be cloned from js2264/BiocBook.template



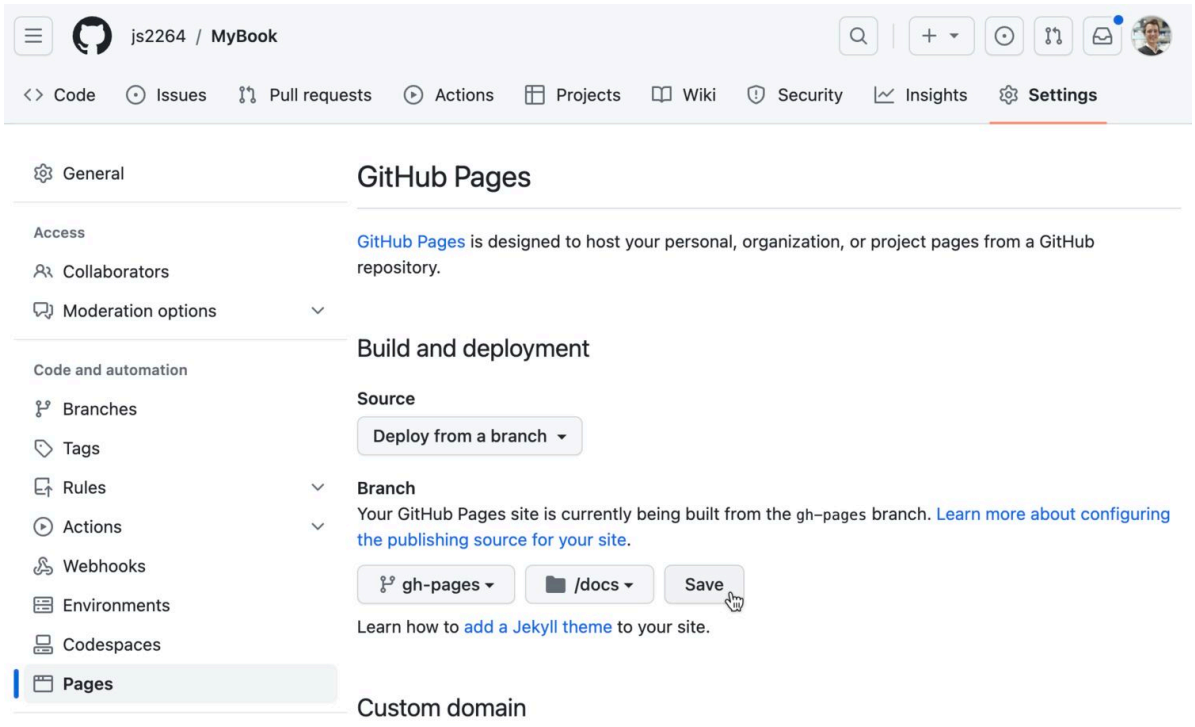
2.2.1.2.2 * Create a new repo



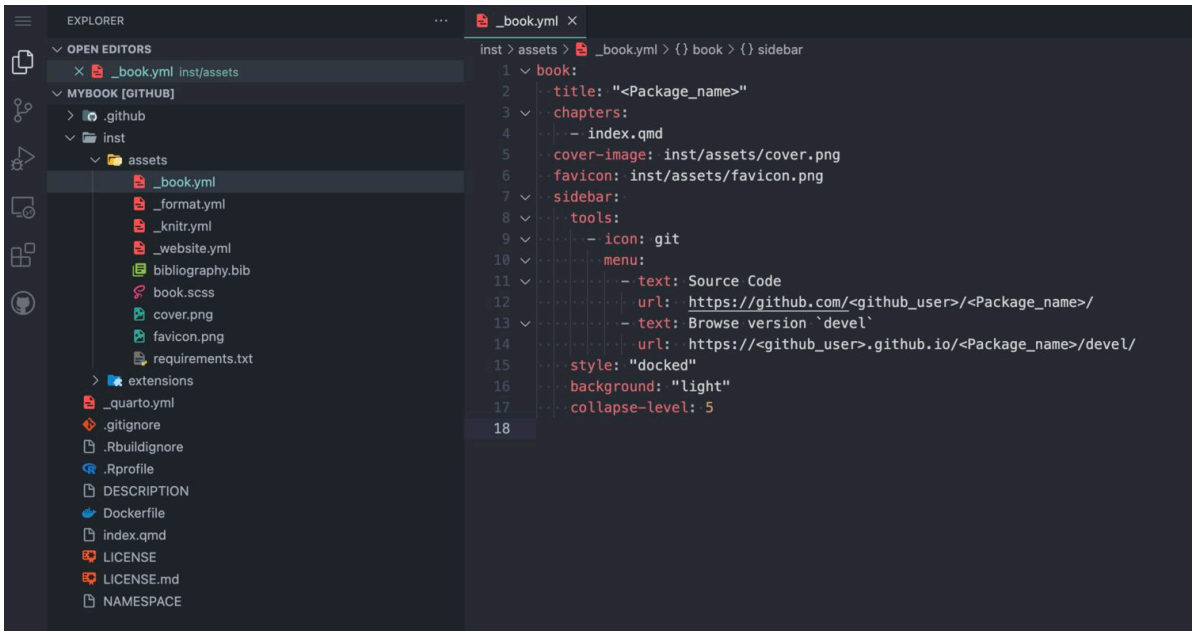
2.2.1.2.3 Enable `Github Pages` to be deployed

You will need to enable the Github Pages service for your newly created repository.

- Go to your new `Github` repository;
- Open the "Settings" tab;
- On the leftside bar, clik on the "Pages" tab;
- Select the `gh-pages` branch and the `/docs` folder to deploy your Github Pages.

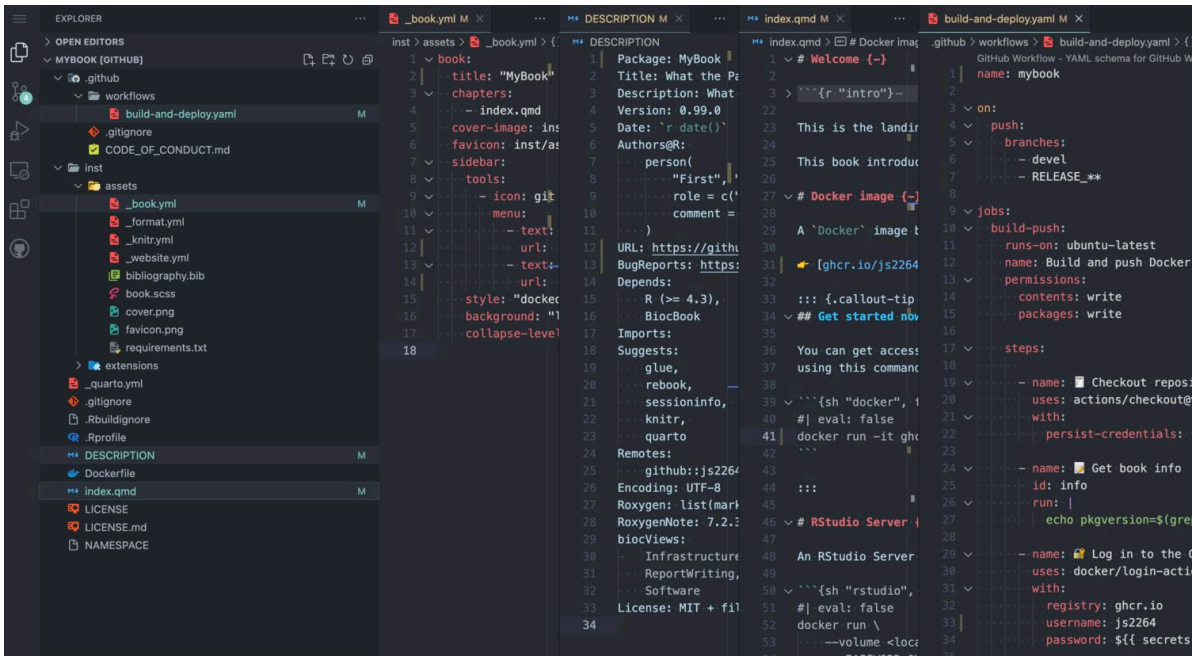


2.2.1.2.4 * Enter VS Code editor by pressing .

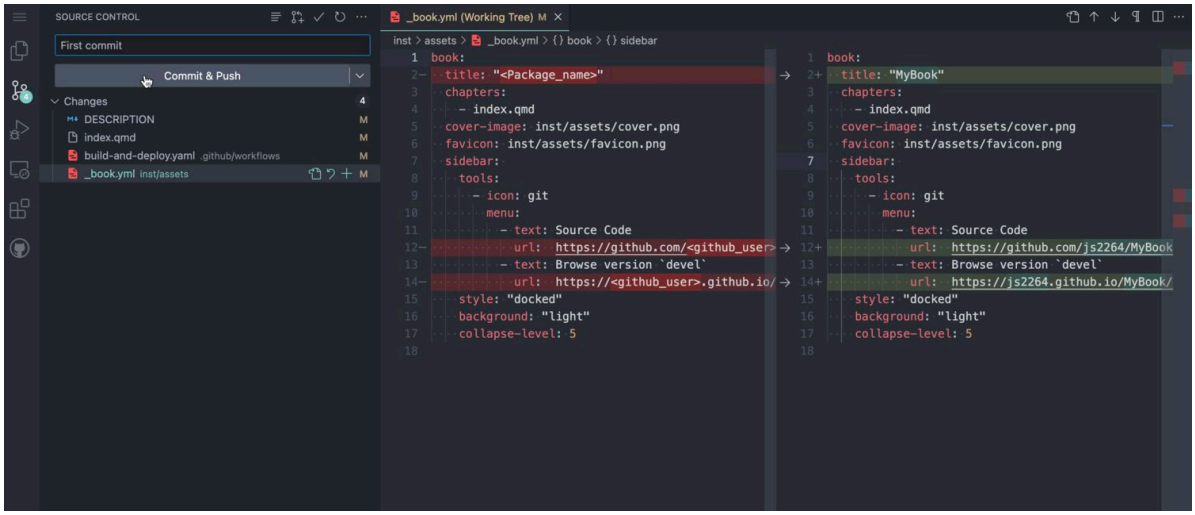


2.2.1.2.5 * Fillout placeholders

> **Warning**
>
> Three types of placeholders need to be replaced:
>
> 1. `<Package_name>`
> 2. `<package_name>`
> 3. `<github_user>`
>
> Three different files contain these placeholders:
>
> 1. `/inst/assets/_book.yml`
> 2. `/DESCRIPTION`
> 3. `/index.qmd`



2.2.1.2.6 * Commit changes



2.2.1.2.7 * Clone the package to a local computer

2.2.2 Edit new `BiocBook` chapters

- `add_chapter(biocbook, title)` is used to write new chapters;
- `add_preamble(biocbook)` is used to add an unnumbered extra page after the Welcome page but before the chapters begin.

> ⚠ Don't forget to add any package used in the book pages to `Imports:` or `Suggests:` fields in `DESCRIPTION`.
>
> This ensures that these packages are installed in the Docker image prior to rendering.

2.2.3 Edit assets

A BiocBook relies on several assets, located in `/inst/assets`.:

- `_book.yml`, `_format.yml`, `_knitr.yml`, `_website.yml`
- `bibliography.bib`
- `book.scss`

To quickly edit these assets, use the corresponding `edit_*` functions:

```
edit_yml(biocbook)
edit_bib(biocbook)
```

```
edit_css(biocbook)
```

2.2.4 Previewing and publishing changes

2.2.4.1 Previewing

While writing, you can monitor the rendering of your book live as follows:

```
preview(biocbook)
```

This will serve a local live rendering of your book.

2.2.4.2 Publishing

Once you are done writing pages of your new book, you should always commit your changes and push them to Github. This can be done as follows:

```
publish(biocbook, message = "  Publish")
```

2.2.4.3 Check your published book and Dockerfiles

This connects to the Github repository associated with a local book and checks the existing branches and Dockerfiles.

```
status(biocbook)
```

2.3 Writing features

2.3.1 Executing code

It's super easy to execute actual code from any `BiocBook` page when rendering the `BiocBook` website.

2.3.1.1 `R` code

`R` code can be executed and rendered:

Listing 2.2 `R`

```
utils::packageVersion("BiocVersion")
## [1] '3.18.0'
```

2.3.1.2 `bash` code

`bash` code can also be executed and rendered:

Listing 2.3 `bash`

```
find ../ -name "*.qmd"
## ../pages/biocbook-vs-rebook.qmd

## ../pages/Chapter-2.qmd


## ../pages/Chapter-1.qmd


## ../pages/preamble.qmd


## ../pages/Chapter-3.qmd


## ../index.qmd
```

2.3.2 Creating data object

While writing chapters, you can save objects as `.rds` files to reuse them in subsequent chapters (e.g. here).

```r
isthisworking <- "yes"
saveRDS(isthisworking, 'isthisworking.rds')
```

2.3.3 Adding references

References can be listed as `.bib` entries in the bibliography file located in `inst/assets/bibliography.bib`. The references can be added in-line using the @ notation, e.g. by typing **@serizay2023**, this will insert the following reference: **@serizay2023**.

3 Containerizing and Publishing against specific Bioconductor release versions

3.1 For a {BiocBook} package not currently in Bioconductor

One can build against older Bioconductor releases as follows:

- Commit current changes before switching branches (should be in `devel` branch)

```
gert::git_commit_all("Commit current changes")
```

- Create a new branch named `RELEASE_X_Y` and checkout

```
gert::git_branch_create("RELEASE_3_17")
```

- Push local `RELEASE_X_Y` to Github

```
gert::git_push()
```

3.2 For a {BiocBook} package accepted in Bioconductor > 6 months ago

Once your package is accepted in Bioconductor, your repository will have access to a new **remote** named `upstream`, pointing to `git@git.bioconductor.org`. When Bioconductor releases a new version, your package will change version on the `upstream` remote, in a dedicated release branch `RELEASE_X_Y`. All details are available here.

To generate a Docker image and a version of the `BiocBook` website built on new Bioconductor releases, you can:

- Add the `Bioconductor` remote to your local repository (this might already be set up)

```
gert::git_remote_add(name = 'upstream', url = 'git@git.bioconductor.org:packages/<YOUR-REP
```

- Create a new local `RELEASE_X_Y` branch

```
gert::git_branch_create("RELEASE_3_15")
```

- Pull the `upstream` `RELEASE_X_Y` commits

```
gert::git_pull("RELEASE_3_15", remote = "upstream")
```

- Push the local `RELEASE_X_Y` branch to `origin` remote (your own Github repository)

```
gert::git_push("RELEASE_3_15", remote = "origin")
```

4 {BiocBook}-based books vs. {rebook}-based books

{rebook} is another book rendering package currently used by Bioconductor to build book packages such as OSCA.* and SingleRBook.

4.1 Differences with {rebook}-based books

OSCA books provided by Bioconductor are rendered upon building by the *Bioconductor Build System* (BBS). They rely on {rebook} to orchestrate the rendering. Briefly, OSCA books have their book pages in /inst/book/, while the vignettes/ folder contains (1) a Makefile and (2) a dummy stub.Rmd vignette (required to trigger vignette rendering and thus make).

When the BBS triggers OSCA.intro package building, upon vignette rendering, the Makefile triggers the following commands:

Listing 4.1 R

```
work.dir <- rebook::bookCache('OSCA.intro')
handle <- rebook::preCompileBook('../inst/book', work.dir=work.dir, desc='../DESCRIPTION')
old.dir <- setwd(work.dir)
bookdown::render_book('index.Rmd')
setwd(old.dir)
rebook::postCompileBook(work.dir=work.dir, final.dir='../inst/doc/book', handle=handle)
```

The resulting book, pre-compiled by {rebook} and assembled by {bookdown}, is eventually served by Bioconductor from the /inst/doc/book/ folder.

{BiocBook}-based packages follow a strategy similar to that of OSCA books: they provide a Makefile in the vignettes/ folder to trigger book rendering when building the package. However, the executed command does not rely on {rebook} and {bookdown}, but on a render command from the quarto software.

The resulting book, fully compiled by native quarto, is also located in /inst/doc/book/ once the package is built (and in doc/book in the library directory once installed).

Listing 4.2 `shell`

```
quarto render ../inst/
mv ../inst/docs ../inst/doc/book
```

⚠ This implies that `quarto` (>= 1.3) has to be installed in the system building the package!

4.2 `BiocBook` features missing from `rebook`

- A `BiocBook` can be readily initiated using `BiocBook::init()`;
- It relies on modern `.qmd` files supported by `Quarto`;
- It can work as a standalone Github-hosted package, without necessarily having to be submitted to/built by Bioconductor. The book should be rendered exactly the same way through Github or by the BBS;
- It supports versioning of the online book served by `gh-pages` **through the author Github account**;
- It distributes versioned Dockerfiles **through the author Github account**;
- `BiocBook`-based packages can actually provide fully-fledged functions in `R/`, manual pages and vignettes. They can be installed exactly the same way than other software packages.

4.3 `rebook` features missing from `BiocBook`

- Smart reuse of objects generated in one book in another book;

Tip

This can still be achieved **within** a book by saving a data object as an `.rds` file and loading it in a subsequent chapter.

```
isthisworking <- readRDS('isthisworking.rds')
isthisworking
##  [1] "yes"
```

- Native support of cross-references across books.