

R Markdown Workshop

- Day 2

Creation, organisation, storage, and collaboration

Anne-Kathrin Kleine | University of Groningen

R Markdown Workshop

- Day 2

Creation, organisation, storage, and collaboration

Anne-Kathrin Kleine | University of Groningen

R Markdown Workshop

- Day 2

Creation, organisation, storage, and collaboration

Anne-Kathrin Kleine | University of Groningen

Version control

Version control



- Research papers have many versions before publication
- two main challenges:
 - keeping track of changes and versions
 - reverting a previous version if necessary
- different approaches for version control:
 - edit, rename and save files
 - use applications (Dropbox, Google Docs, Overleaf)
 - use version control systems such as Git and GitHub

```
manuscript
|
|- journals_FINAL_19May.Rmd
|- journals_PAPERVERS_17May.Rmd
|- journals_10May.Rmd
|- journals_FINAL.Rmd
|- journals_26APRIL_newliterature.Rmd
...
|
|- journals.Rproj
|- references.bib
|- apa_7th.csl
```

Version control with Git and GitHub

Advantages of using Git and GitHub

🎈 Contribute to open source projects, see [here](#)

- almost all open source projects use GitHub

🎤 Showcase your work

🔍 Track changes across versions

👩‍💻 👨‍💻 Collaborate on projects

👨‍💻 Various integration options

- Amazon and Google Cloud; or use GitHub pages or Netlify to build websites based on your R files

Version Control – Git and GitHub – Definitions



- a software that keeps track of versions of a set of files
- it is *local* to you; the records are kept on your computer

Version Control – Git and GitHub – Definitions



- a software that keeps track of versions of a set of files
- it is *local* to you; the records are kept on your computer



- a hosting service that can keep the records
- it is *remote* to you, like Dropbox
- GitHub is specifically structured to keep records with Git

Version Control — Git and GitHub — Definitions

Repository, or repo

- a set of files whose records are kept together, by Git and/or on GitHub

Version Control – Git and GitHub – Definitions

Repository, or repo

- a set of files whose records are kept together, by Git and/or on GitHub

To commit

- to take a snapshot of a repository
- it is local, the records are kept on your computer unless you push

Version Control – Git and GitHub – Definitions

Repository, or repo

- a set of files whose records are kept together, by Git and/or on GitHub

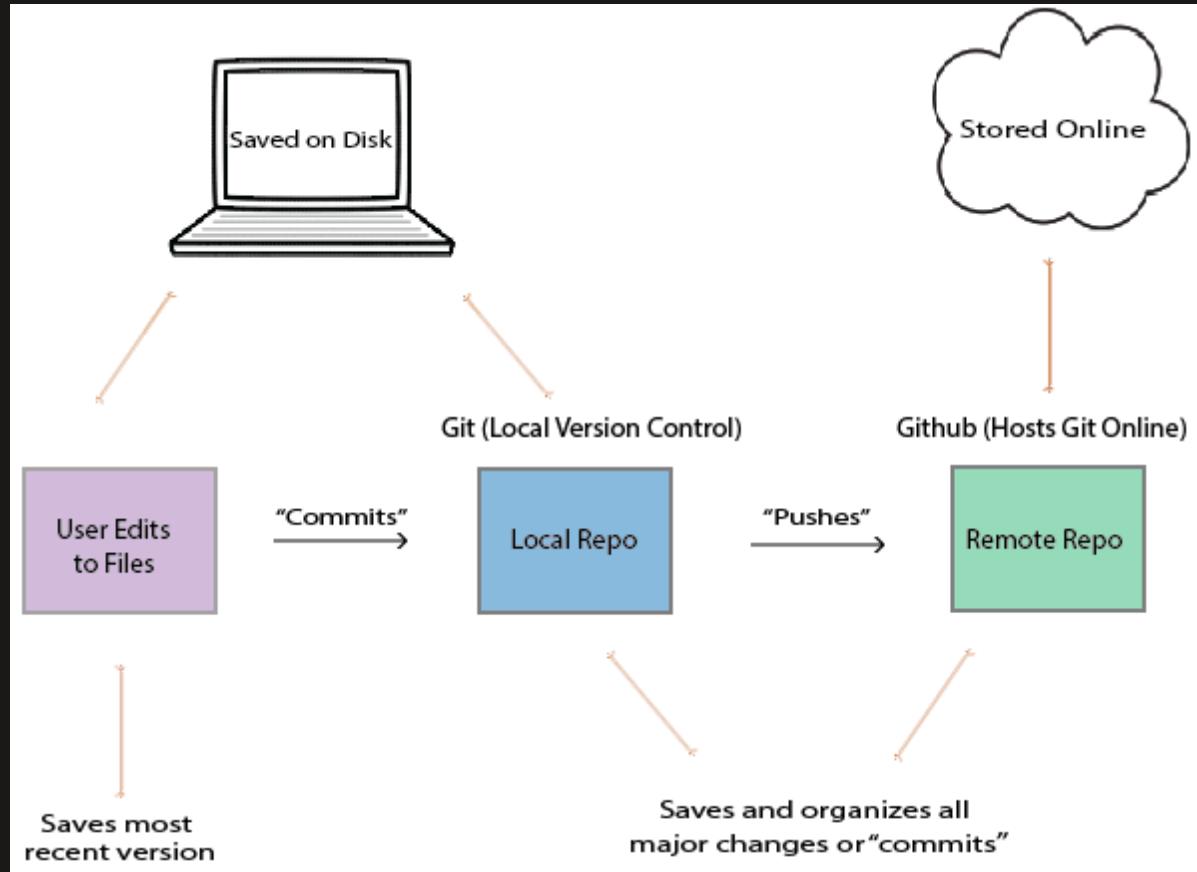
To commit

- to take a snapshot of a repository
- it is local, the records are kept on your computer unless you push

To push

- to move a copy of the records from Git to GitHub, from your computer to online server
- it is like uploading (the new versions of) your files and sub-folders to a website

Version Control – Git and GitHub



Source: <https://iqss.github.io/dss-rbuild/version-control.html>

How familiar are you with
using the command line
(terminal, power shell...)?

Go to <https://www.menti.com/9eof4daf4h>  (and enter the
code 4173 2433)

[See results](#)

Version Control – Git and GitHub – check setup

In the 🖥 terminal 🖥 :git --version

- you should see the Git version ✓

In the 🖥 terminal 🖥 :git config --global user.name

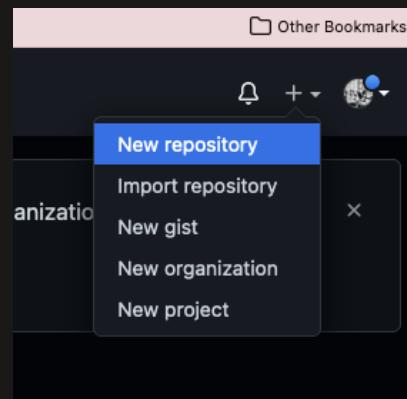
- you should see the Git user name ✓

🚫 If these steps did not work, go back [here](#) and continue setting up Git where you left off

You really don't want to use the terminal ? ▶ For version control in RStudio, see [here](#), and particularly [this video](#)

Version Control – Git and GitHub – connect local with remote

1. Go to GitHub and create a new repository



2. Fill in some info, create a public repository (don't bother with Readme etc - those will be added later!)

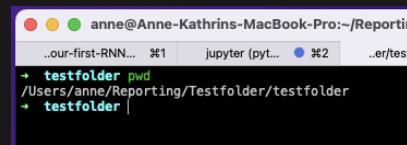
A screenshot of the 'Create a new repository' form on GitHub. The form has a dark background. At the top, it says 'Create a new repository' and explains what a repository is. It asks for 'Repository template' (No template selected). Under 'Owner *', it shows 'AnneOkk / testrepo'. It says 'Great repository names like testrepo are available. Need inspiration? How about curly-octo-pancake?'. There's a 'Description (optional)' field which is empty. Below that, there are two radio buttons: 'Public' (selected) and 'Private'. It says 'Anyone on the internet can see this repository. You choose who can commit.' for Public and 'You choose who can see and commit to this repository.' for Private. Under 'Initialize this repository with:', there's a checkbox for 'Add a README file' (unchecked) with a note: 'This is where you can write a long description for your project. Learn more.' Below that is a section for 'Add .gitignore' with a note: 'Choose which files not to track from a list of templates. Learn more.' and a dropdown for 'gitignore template: None'. At the bottom, there's a section for 'Choose a license' with a note: 'A license tells others what they can and can't do with your code. Learn more.' and a dropdown for 'License: None'.

Version Control – Git and GitHub – connect local with remote

3. Follow the steps in Option 1: "...create a new repository on the command line"

The screenshot shows a GitHub repository page for "AnneOkk/testrepo". The top navigation bar includes "Pin", "Unwatch 1", "Fork", "Star 0", and a dropdown menu. Below the header, there are tabs for "Code", "Issues", "Pull requests", "Actions", "Projects", "Wiki", "Security", and "Insights". A large blue box highlights the "Quick setup — if you've done this kind of thing before" section. It contains instructions for setting up the repository via "Set up in Desktop" (with icons for GitHub, Google Drive, OneDrive, and Dropbox), "HTTPS", or "SSH", and provides the URL "https://github.com/AnneOkk/testrepo.git". Below this, there's a note about including a README, LICENSE, and .gitignore file. Further down, sections for "...or create a new repository on the command line" and "...or push an existing repository from the command line" provide specific git commands. At the bottom, there's a section for "...or import code from another repository" with a "Import code" button.

4. In the terminal , navigate to your R project folder



Version Control – Git and GitHub – connect local with remote

5. In the terminal , type:

`git init` this initializes a git repo on your local machine

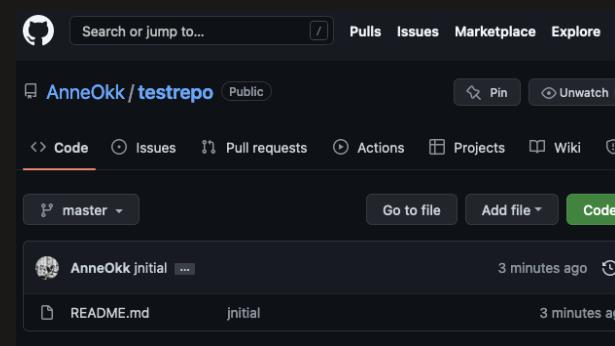
`touch README.md` this creates an empty README file

`git add .` this stages all the content in that folder to be committed

`git commit -m "add empty readme"` stages all content to be pushed

```
git remote add origin  
https://github.com/AnneOkk/testrepo.git substitute  
with your repo URL
```

```
git push origin master
```

 push all the content from Git to GitHub

Version Control – Git and GitHub – connect local with remote

5. In the terminal, type:

`git init` this initializes a git repo on your local machine

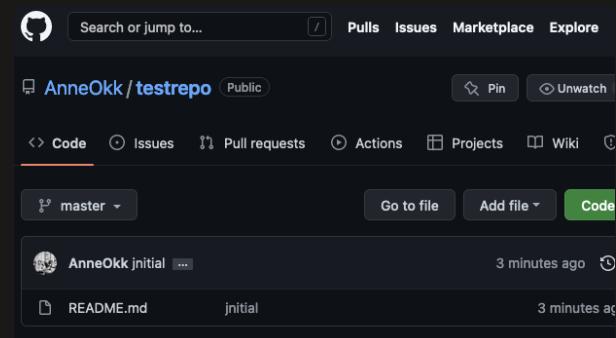
`touch README.md` this creates an empty README file

`git add .` this stages all the content in that folder to be committed

`git commit -m "add empty readme"` stages all content to be pushed

`git remote add origin https://github.com/AnneOkk/testrepo.git` substitute with your repo URL

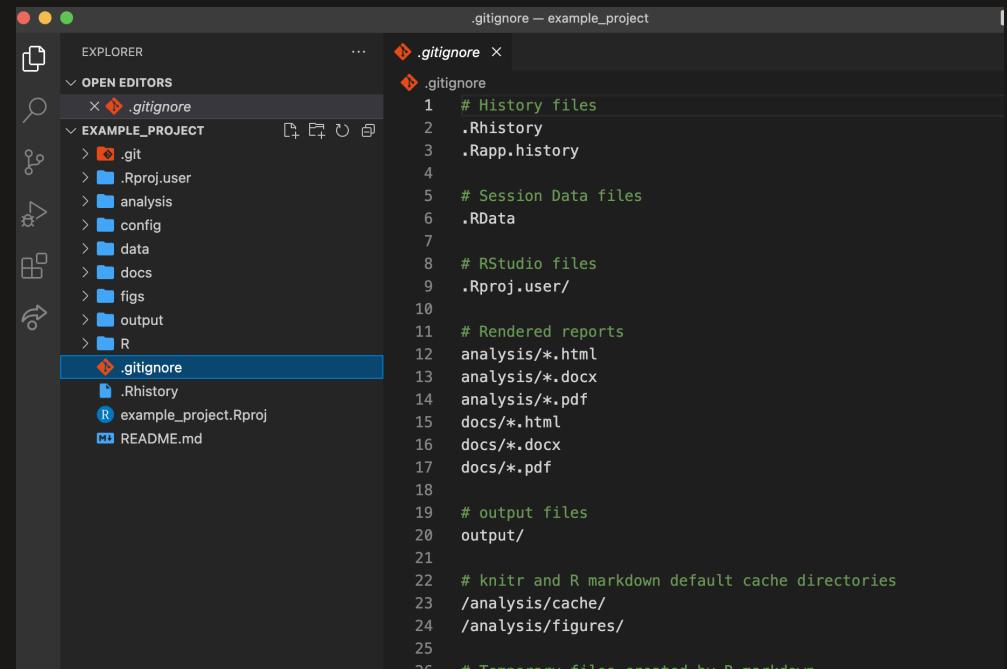
`git push origin master` push all the content from Git to GitHub



Pushed files should appear in GitHub repository

Version Control – Git and GitHub – .gitignore

- `.gitignore` file specifies which file(s) and/or folder(s) should be excluded from version control
- lists one item per line; each line has a pattern, which determines whether files should be ignored
- see [gitignore documentation](#) for more info on the different patterns
- reasons to exclude files from being pushed include:
 - personal information others should not see (e.g., credential files)
 - no rights to share (e.g., certain data files)
 - outputs that may be re-created (e.g., `outputs.pdf`)



```
.gitignore -- example_project
.gitignore
1 # History files
2 .Rhistory
3 .Rapp.history
4
5 # Session Data files
6 .RData
7
8 # RStudio files
9 .Rproj.user/
10
11 # Rendered reports
12 analysis/*.html
13 analysis/*.docx
14 analysis/*.pdf
15 docs/*.html
16 docs/*.docx
17 docs/*.pdf
18
19 # output files
20 output/
21
22 # knitr and R markdown default cache directories
23 /analysis/cache/
24 /analysis/figures/
25
26 # Temporary files created by R markdown
```

Version Control — Git and GitHub — collaborating

Imagine... 🤔

Version Control — Git and GitHub — collaborating

Imagine... 🤔

...you work in a team of data scientists/ data enthusiastic researchers 🧑🏿‍💻🧙🏿‍♂️🧙🏿‍♀️🧙🏿‍♂️🧙🏿‍♀️

Version Control — Git and GitHub — collaborating

Imagine... 🤔

...you work in a team of data scientists/ data enthusiastic researchers 

...you want to be able to work together on a data analysis project 

Version Control — Git and GitHub — collaborating

Imagine... 🤔

...you work in a team of data scientists/ data enthusiastic researchers 👨‍💻🧙‍♂️🧙‍♀️🧙‍♂️🧙‍♀️

...you want to be able to work together on a data analysis project 👨‍💻👩‍💻👩‍💻👨‍💻👩‍💻

...you want to keep track of what the others do and finally merge individual contributions into one overall project ✨💾✨

Version Control – Git and GitHub – collaborating

Imagine... 🤔

...you work in a team of data scientists/ data enthusiastic researchers 👨‍🧙‍🧙‍🧙‍🧙‍🧙

...you want to be able to work together on a data analysis project 💻💻💻💻

...you want to keep track of what the others do and finally merge individual contributions into one overall project ✨💻✨

But how ? 🤔

Version Control – Git and GitHub – collaborating

Scenario 1 - you are owner/ collaborator

... first things first!

Version Control – Git and GitHub – collaborating

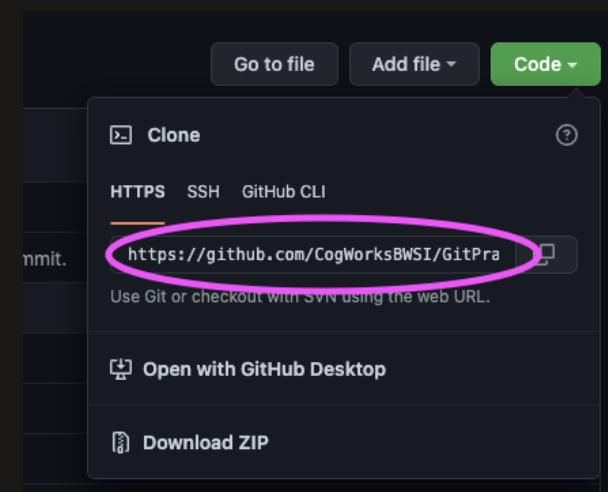
Scenario 1 - you are owner/ collaborator

... first things first!

1) Go to the GitHub repository you are working on as a team

2) In the terminal , navigate to where you want to store the project files and type:

```
git clone https://github.com/Anne0kk/testrepo.git
```



Version Control – Git and GitHub – collaborating

3) Next, you would always want to make sure that you are up to date with what has changed in the remote repository

Version Control — Git and GitHub — collaborating

3) Next, you would always want to make sure that you are up to date with what has changed in the remote repository

In the  terminal  , type:

```
git pull origin master
```

Version Control — Git and GitHub — collaborating

3) Next, you would always want to make sure that you are up to date with what has changed in the remote repository

In the  terminal  , type:

```
git pull origin master
```

4) Create a new branch to add changes that belong to a novel feature you are working on (this prevents messing up the master branch)

```
git checkout -b "cool-feature"
```

Version Control – Git and GitHub – collaborating

3) Next, you would always want to make sure that you are up to date with what has changed in the remote repository

In the  terminal  , type:

```
git pull origin master
```

4) Create a new branch to add changes that belong to a novel feature you are working on (this prevents messing up the master branch)

```
git checkout -b "cool-feature"
```

5) Now make some changes (e.g., add the new cool feature)

Version Control – Git and GitHub – collaborating

3) Next, you would always want to make sure that you are up to date with what has changed in the remote repository

In the  terminal  , type:

```
git pull origin master
```

4) Create a new branch to add changes that belong to a novel feature you are working on (this prevents messing up the master branch)

```
git checkout -b "cool-feature"
```

5) Now make some changes (e.g., add the new cool feature)

6) Add, commit, and push the files ...through the usual workflow (`git add .,git commit -m "new feature file added"`)



Version Control — Git and GitHub — collaborating

3) Next, you would always want to make sure that you are up to date with what has changed in the remote repository

In the  terminal  , type:

```
git pull origin master
```

4) Create a new branch to add changes that belong to a novel feature you are working on (this prevents messing up the master branch)

```
git checkout -b "cool-feature"
```

5) Now make some changes (e.g., add the new cool feature)

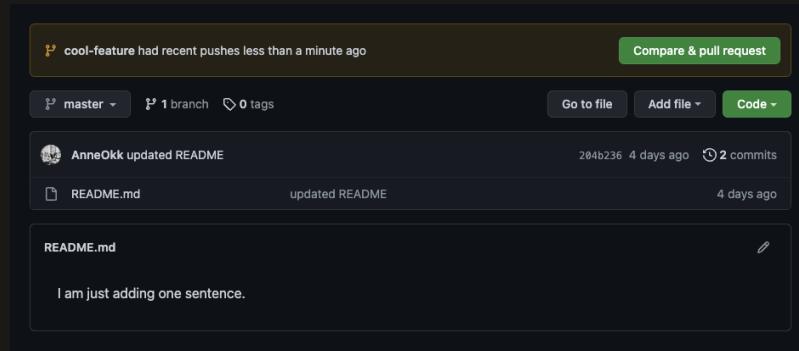
6) Add, commit, and push the files ...through the usual workflow (`git add ., git commit -m "new feature file added"`)



  BUT you push the changes to the newly created branch to prevent messing up the master branch!  

```
git push origin cool-feature
```

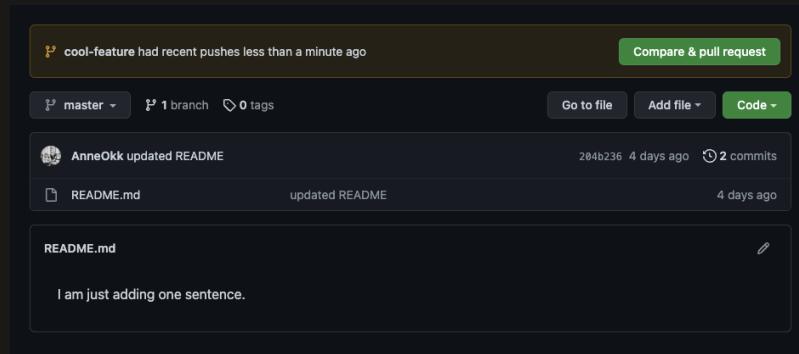
Version Control – Git and GitHub – collaborating



7) On GitHub, you create a pull request

Reviewers (other team members) may be assigned to check the changes and start a conversation (through comments)

Version Control – Git and GitHub – collaborating



7) On GitHub, you create a pull request

Reviewers (other team members) may be assigned to check the changes and start a conversation (through comments)

8) Once all issues are resolved, your changes will be merged into the master branch 😊

Version Control – Git and GitHub – collaborating

... But what if some issues remain unresolved? 🤪 "You forgot to read in the data!" 🤔

Version Control – Git and GitHub – collaborating

... But what if some issues remain unresolved? 🤔 "You forgot to read in the data!" 🤔

If the branch has not been removed, you may add smaller changes (on the same feature) through `git add ., git commit -m "added data reader", git push origin branch-name`

Version Control — Git and GitHub — collaborating

... But what if some issues remain unresolved? 🤔 "You forgot to read in the data!" 🤦

If the branch has not been removed, you may add smaller changes (on the same feature) through `git add ., git commit -m "added data reader", git push origin branch-name`

! BUT if some time has passed and/or the addition is major !

Version Control – Git and GitHub – collaborating

... But what if some issues remain unresolved? 🤦 "You forgot to read in the data!" 🤔

If the branch has not been removed, you may add smaller changes (on the same feature) through `git add ., git commit -m "added data reader", git push origin branch-name`

! BUT if some time has passed and/or the addition is major !

▶ Checkout to the master branch:

```
git checkout master
```

Version Control — Git and GitHub — collaborating

... But what if some issues remain unresolved? 🤦 "You forgot to read in the data!" 🤔

If the branch has not been removed, you may add smaller changes (on the same feature) through `git add ., git commit -m "added data reader", git push origin branch-name`

! BUT if some time has passed and/or the addition is major !

▶ Checkout to the master branch:

```
git checkout master
```

Pull all changes that happened in the mean time:

```
git pull origin master
```

Version Control – Git and GitHub – collaborating

... But what if some issues remain unresolved? 🤦 "You forgot to read in the data!" 🤪

If the branch has not been removed, you may add smaller changes (on the same feature) through `git add ., git commit -m "added data reader", git push origin branch-name`

! BUT if some time has passed and/or the addition is major !

▶ Checkout to the master branch:

```
git checkout master
```

Pull all changes that happened in the mean time:

```
git pull origin master
```

Checkout a new branch for the missing feature;

```
git checkout -b data-reader
```

Version Control – Git and GitHub – collaborating

... But what if some issues remain unresolved? 🤦 "You forgot to read in the data!" 🤪

If the branch has not been removed, you may add smaller changes (on the same feature) through `git add ., git commit -m "added data reader", git push origin branch-name`

! BUT if some time has passed and/or the addition is major !

▶ Checkout to the master branch:

```
git checkout master
```

Pull all changes that happened in the mean time:

```
git pull origin master
```

Checkout a new branch for the missing feature;

```
git checkout -b data-reader
```

Make the necessary changes!

Version Control – Git and GitHub – collaborating

... But what if some issues remain unresolved? 🤦 "You forgot to read in the data!" 😱

If the branch has not been removed, you may add smaller changes (on the same feature) through `git add ., git commit -m "added data reader", git push origin branch-name`

! BUT if some time has passed and/or the addition is major !

▶ Checkout to the master branch:

```
git checkout master
```

Pull all changes that happened in the mean time:

```
git pull origin master
```

Checkout a new branch for the missing feature;

```
git checkout -b data-reader
```

Make the necessary changes!

... And then, the usual workflow (don't forget to push to new branch!)

Version Control – Git and GitHub – collaborating

Scenario 2 - you want to contribute to an open science project

Version Control – Git and GitHub – collaborating

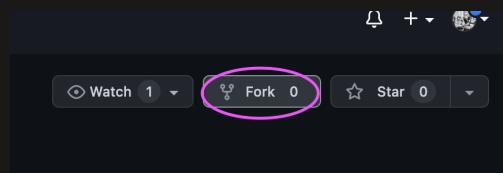
Scenario 2 - you want to contribute to an open science project

- 1) In GitHub, navigate to the project you want to contribute to

Version Control – Git and GitHub – collaborating

Scenario 2 - you want to contribute to an open science project

- 1) In GitHub, navigate to the [project](#) you want to contribute to
- 2) Fork the repo!

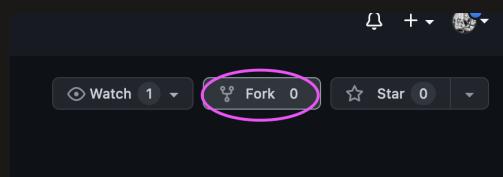


Version Control – Git and GitHub – collaborating

Scenario 2 - you want to contribute to an open science project

1) In GitHub, navigate to the project you want to contribute to

2) Fork the repo!



3) Clone the new repo!

```
git clone https://github.com/Anne0kk/Testrepo-1.git
```

4) Make your changes and then add, commit, and push to your master branch

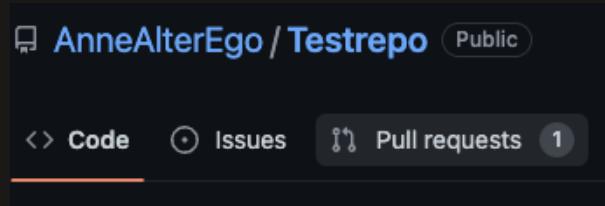
4) Make your changes and then add, commit, and push to your master branch

5) On GitHub, go back to the original repository and create a pull request.

You may need to click "compare across forks"

The screenshot shows a GitHub interface for comparing changes between two repositories. At the top, it says "Comparing changes" and "Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks." Below this, there are dropdown menus for "base repository" (AnneAlterEgo/Testrepo), "base" (main), "head repository" (AnneOkk/Testrepo-1), and "compare" (main). A green checkmark indicates "Able to merge. These branches can be automatically merged." A "Create pull request" button is visible. The main content area shows "1 commit", "1 file changed", and "1 contributor". It lists a commit from "Commits on May 18, 2022" by "AnneOkk" at 6d4a739, which added a new feature. Below this, it says "Showing 1 changed file with 0 additions and 0 deletions." The file listed is "new_feat.R", which is described as an "Empty file."

... On the other side

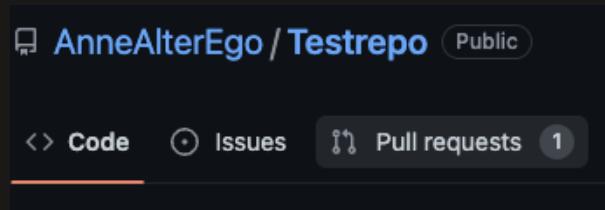


Open and review the pull request! 🔎

If you're happy with the changes, merge! 👍

Otherwise, request additional changes in comments. 💬

... On the other side



Open and review the pull request! 🔎

If you're happy with the changes, merge! 👍

Otherwise, request additional changes in comments. 💬

🎉 Yay, you're all set to contribute to open science projects on GitHub! 🎉

Your turn!



1. Connect your local R project folder with a GitHub repository
2. Change some of the content in R, save, and then push the changes to GitHub
 - `git add .`
 - `git commit -m "senseful commit message that describes the change(s)"`
 - `git push origin master`
3. [OPTIONAL] Create a pull request for your own repository and merge the changes into the master branch
4. [OPTIONAL] One team member creates a pull request for the other team member's repository (first fork and clone); the repo owner merges the changes into her/his master (or main) branch.

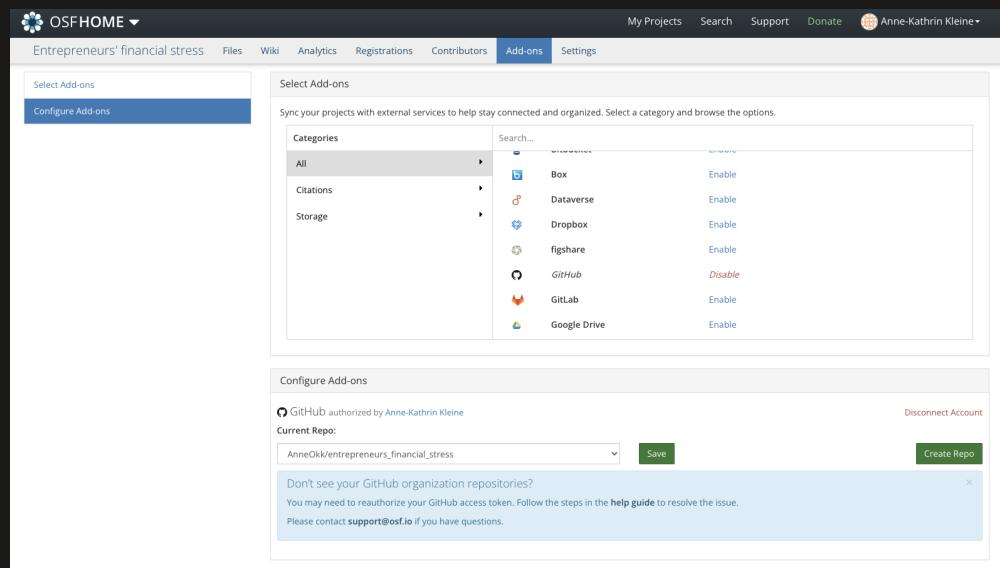
Integrating osf with GitHub



Integrating osf with GitHub

1. Create your osf project

2. Enable GitHub in Add-ons



The screenshot shows the 'Add-ons' section of the OSF interface. In the 'Select Add-ons' panel, 'GitHub' is listed under the 'Storage' category with the status 'Disable'. Below this, the 'Configure Add-ons' panel shows a GitHub account authorized by 'Anne-Kathrin Kleine'. A message at the bottom indicates that the GitHub organization repositories are not visible.

Select Add-ons

Configure Add-ons

GitHub authorized by Anne-Kathrin Kleine

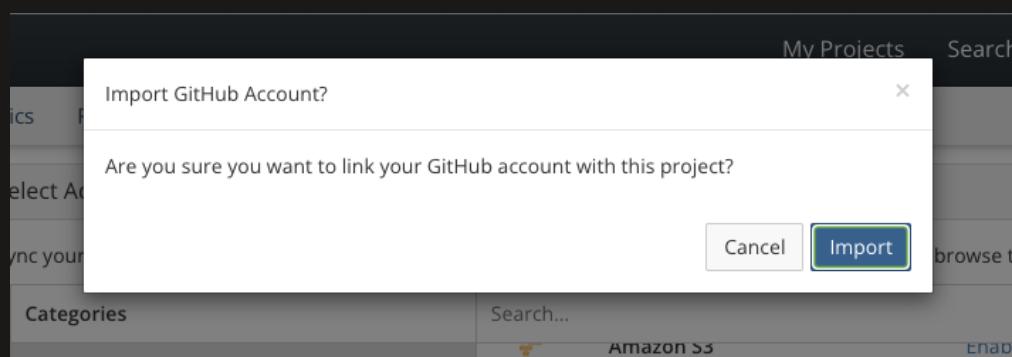
Current Repo: AnneOkk/entrepreneurs_financial_stress

Save Create Repo

Don't see your GitHub organization repositories? You may need to reauthorize your GitHub access token. Follow the steps in the [help guide](#) to resolve the issue. Please contact support@osf.io if you have questions.

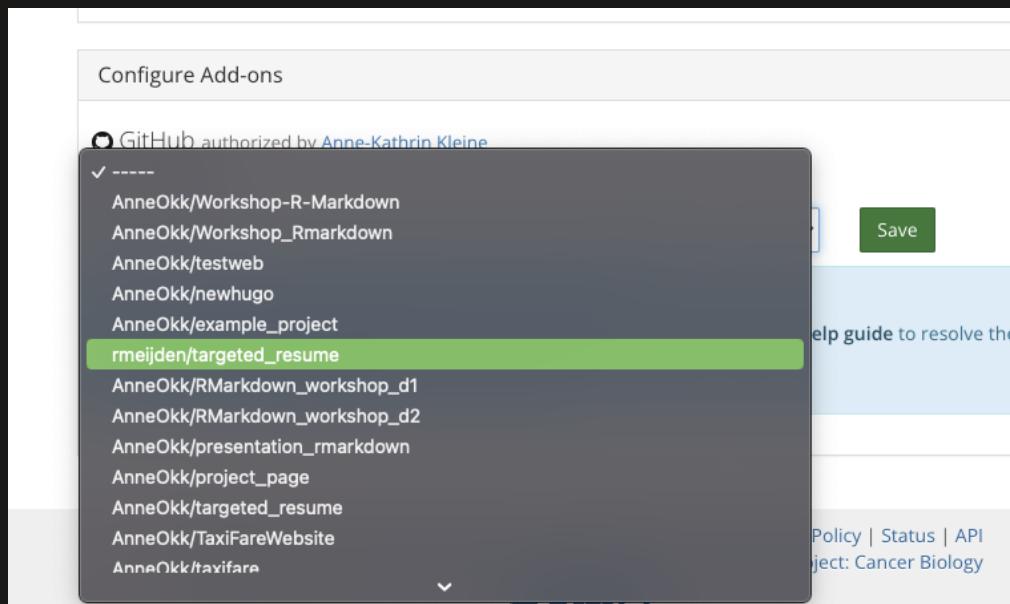
Integrating osf with GitHub

3. Import GitHub Account



Integrating osf with GitHub

4. Select Repo



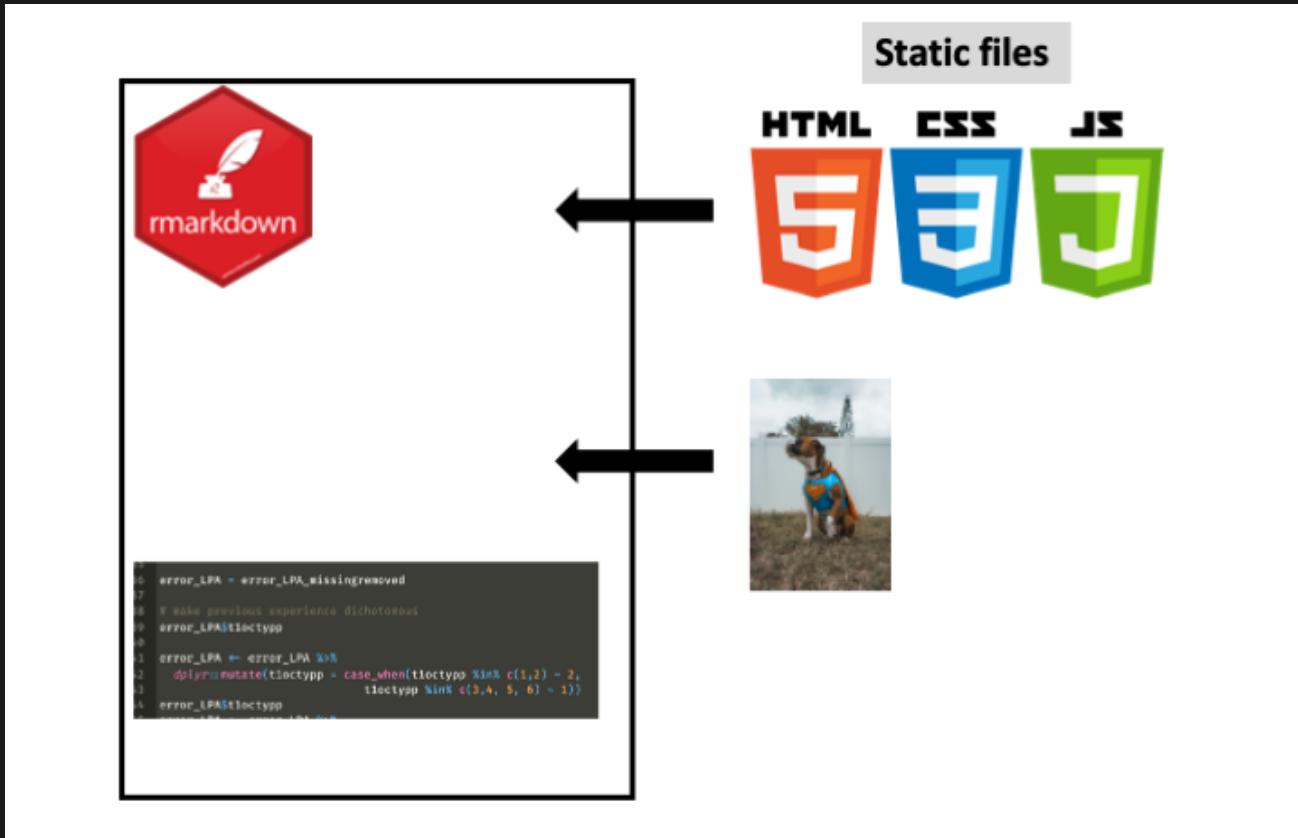
Meet *blogdown*



Making Websites in R Markdown

Major resources: [Alison Hill and Yihui Xie's Advanced R Workshop](#)

What is blogdown?



What is blogdown?

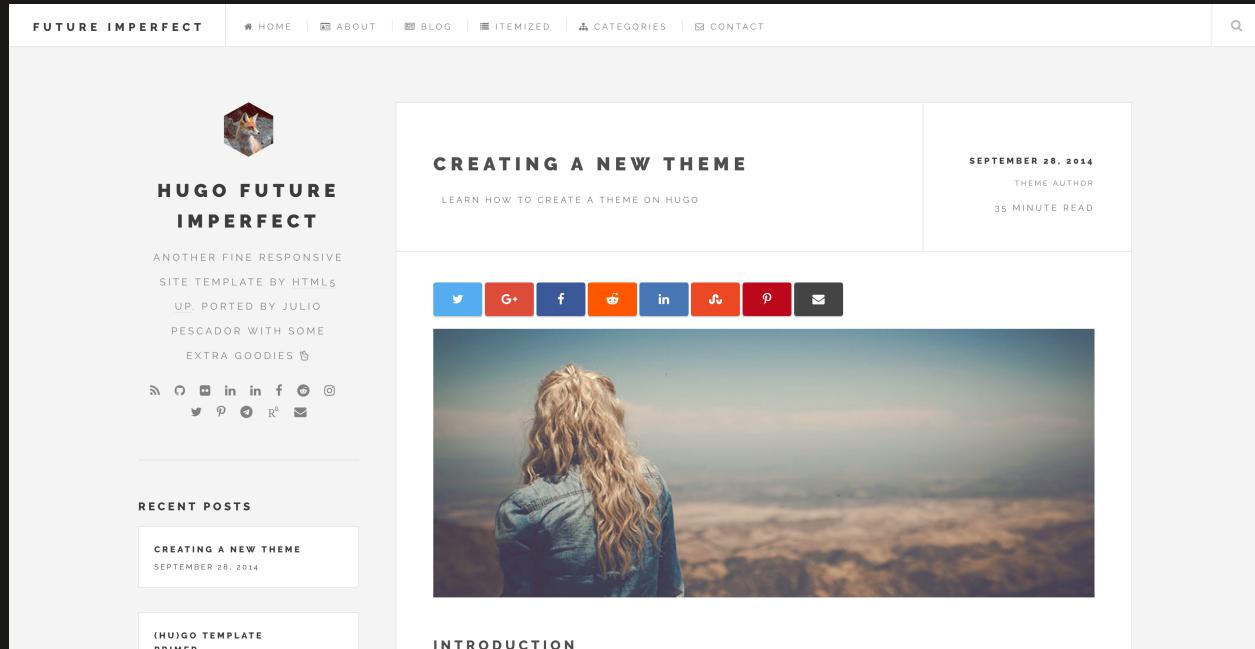


- Hugo is a static site generator
- free, open-source, relatively easy to use
- extremely fast and multiple purpose (websites, blogs,...)

Why not WordPress, Tumblr, Medium.com, Blogger.com, etc?

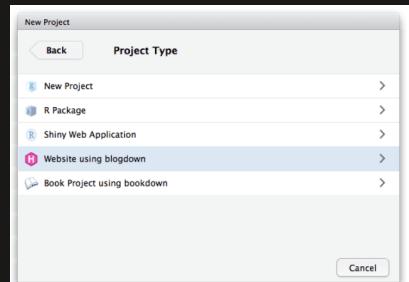
- No R (Markdown) support (even math support is often nonexistent)
- If there is anything related to R, statistical computing, and/or graphics, blogdown will be much more convenient

Build a site for your project



Build a site for your project

1) Create an R blogdown project



2) Make a [repo on GitHub](#)

3) Connect remote repo to local

Build a site for your project

4) Serve site

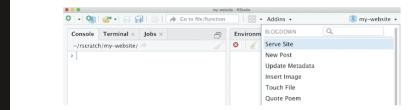
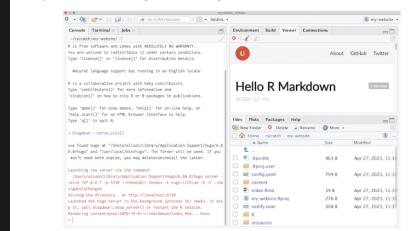


FIGURE 1.3: Use the blogdown addin to serve the site.

Then you should see something that looks like Figure 1.4.





Gaze into your directory structure



Directory Structure

```
.  
├── R  
├── config.yaml  
├── content  
├── website.Rproj  
├── index.Rmd  
├── layouts  
├── netlify.toml  
├── public  
├── resources  
├── static  
└── themes
```



Directory Structure

```
.  
├── R  
├── config.yaml  
├── content  
├── website.Rproj  
└── index.Rmd # DO NOT TOUCH!  
├── layouts  
├── netlify.toml  
├── public  
├── resources  
├── static  
└── themes
```



Directory Structure

```
.  
├── R  
├── config.yaml  
├── content  
├── website.Rproj  
└── index.Rmd # DO NOT TOUCH!  
├── layouts  
├── netlify.toml  
├── public  
├── resources  
├── static  
└── themes
```

```
---  
site: blogdown:::blogdown_site  
---
```



Directory Structure

```
.  
├── R  
├── config.yaml  
├── content  
├── website.Rproj  
├── index.Rmd  
├── layouts  
├── netlify.toml  
└── public # IGNORE!  
├── resources  
├── static  
└── themes
```

DON'T  KNIT!





Serve site!

Mouse up to "Addins" ➔ "Serve site"



Success?

FUTURE IMPERFECT

HOME | ABOUT | BLOG | ITEMIZED | CATEGORIES | CONTACT

Q



HUGO FUTURE
IMPERFECT

ANOTHER FINE RESPONSIVE
SITE TEMPLATE BY HTML5
UP. PORTED BY JULIO
PESCADOR WITH SOME
EXTRA GOODIES

RSS G+ f o in in f o o
Twitter Google+ Facebook YouTube Instagram
RSS GitHub LinkedIn Facebook YouTube Instagram

RECENT POSTS

CREATING A NEW THEME
SEPTEMBER 28, 2014

(HUGO) TEMPLATE PRIMER

CREATING A NEW THEME

LEARN HOW TO CREATE A THEME ON HUGO

SEPTEMBER 28, 2014

THEME AUTHOR

35 MINUTE READ



INTRODUCTION

"Show in new window" to see in local browser

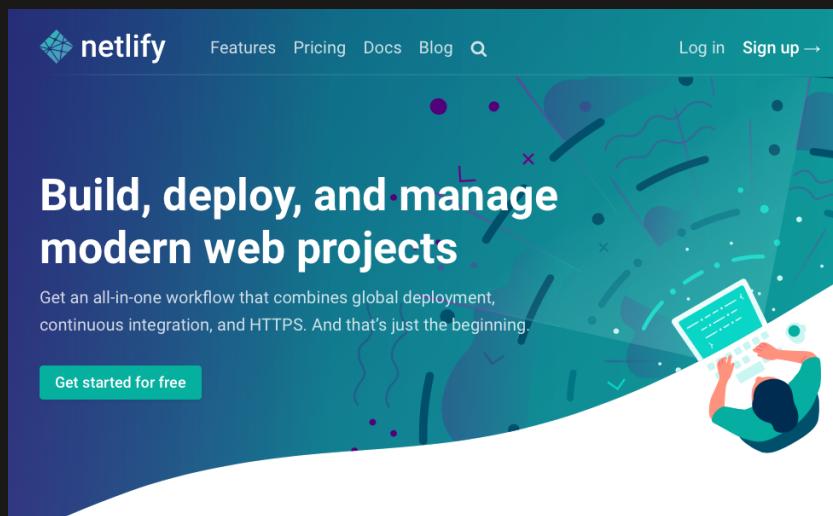
Publishing



Publishing

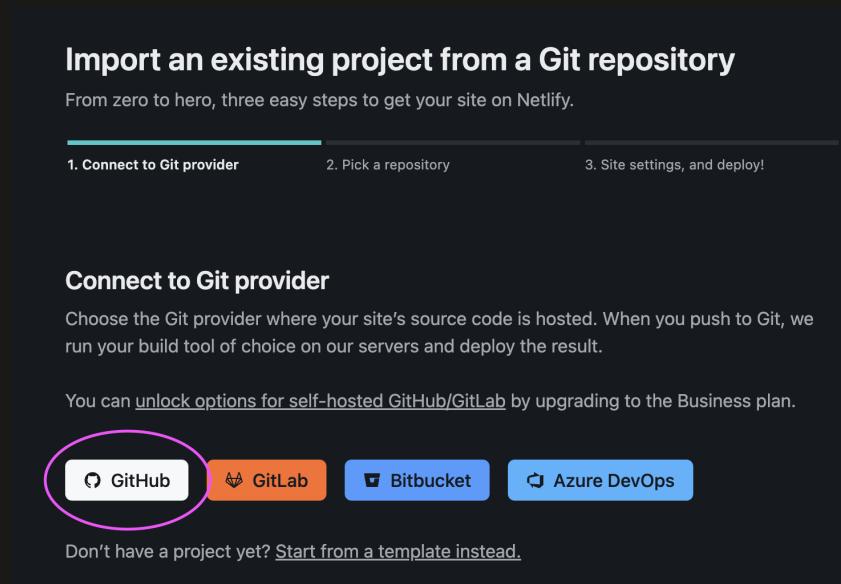


Go to Netlify.com & Log in



Publishing

Connect netlify with to GitHub repository

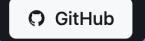
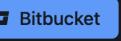


Import an existing project from a Git repository
From zero to hero, three easy steps to get your site on Netlify.

1. Connect to Git provider 2. Pick a repository 3. Site settings, and deploy!

Connect to Git provider
Choose the Git provider where your site's source code is hosted. When you push to Git, we run your build tool of choice on our servers and deploy the result.

You can unlock options for self-hosted GitHub/GitLab by upgrading to the Business plan.

 GitHub  GitLab  Bitbucket  Azure DevOps

Don't have a project yet? [Start from a template instead](#).

Publishing 📢

Configure Netlify on GitHub

Import an existing project from a Git repository

From zero to hero, three easy steps to get your site on Netlify.

1. Connect to Git provider 2. Pick a repository 3. Site settings, and deploy!

Pick a repository from GitHub

Choose the repository you want to link to your site on Netlify. When you push to Git, we run your build tool of choice on our servers and deploy the result.

 AnneOkk ▾

No repositories found

This can happen when Netlify doesn't have access to the repositories in an account. Configure the Netlify app on GitHub, and give it access to the repository you want to link.

[Configure Netlify on GitHub](#)

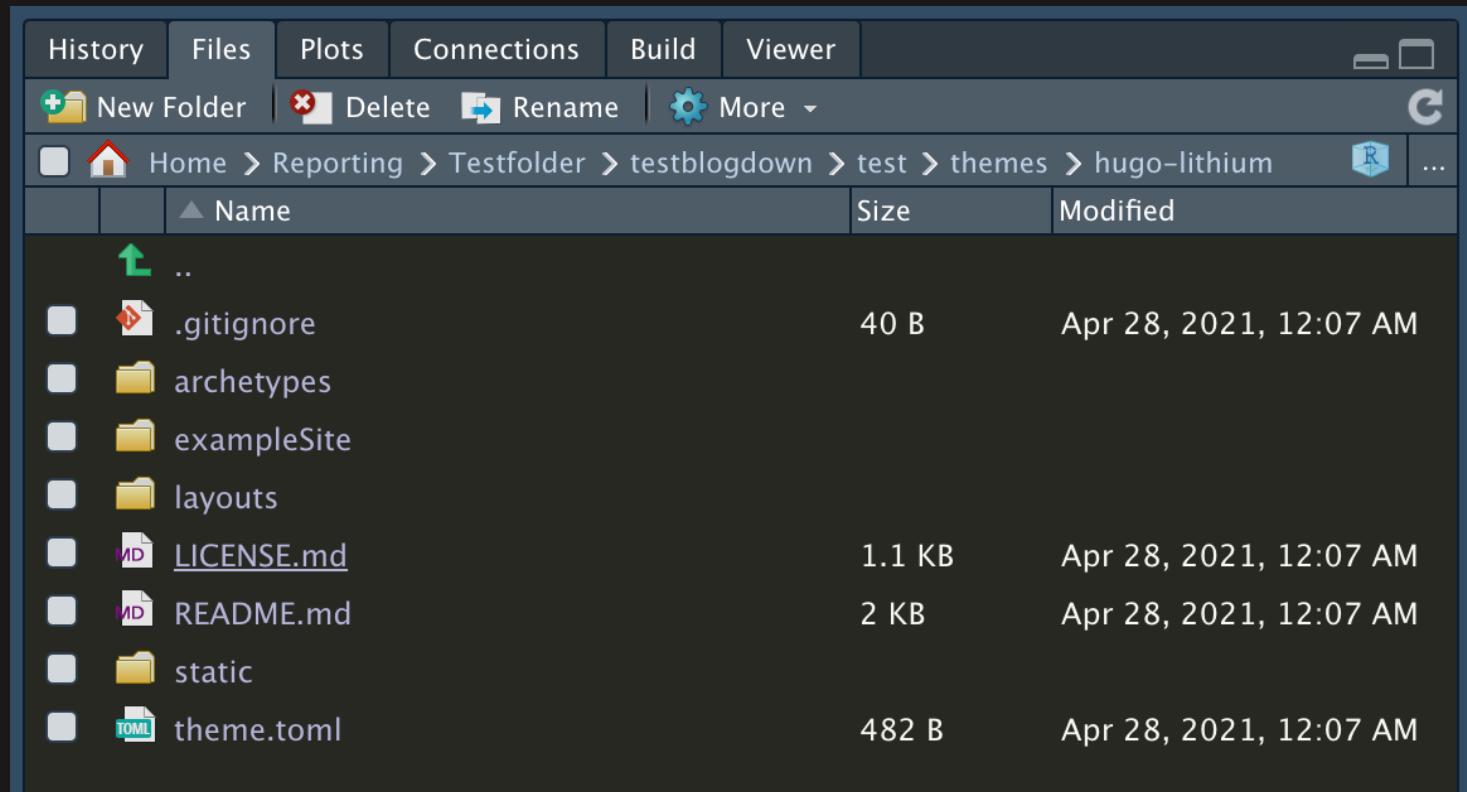


Directory Structure

```
.  
├── R  
├── config.yaml  
├── content  
├── website.Rproj  
├── index.Rmd  
├── layouts  
├── netlify.toml  
├── public  
├── resources  
├── static  
└── themes # Do NOT TOUCH!
```

Themes

Look in `/themes/hugo-lithium/`. Notice now the folder structure here *mirrors* your Hugo directory structure.



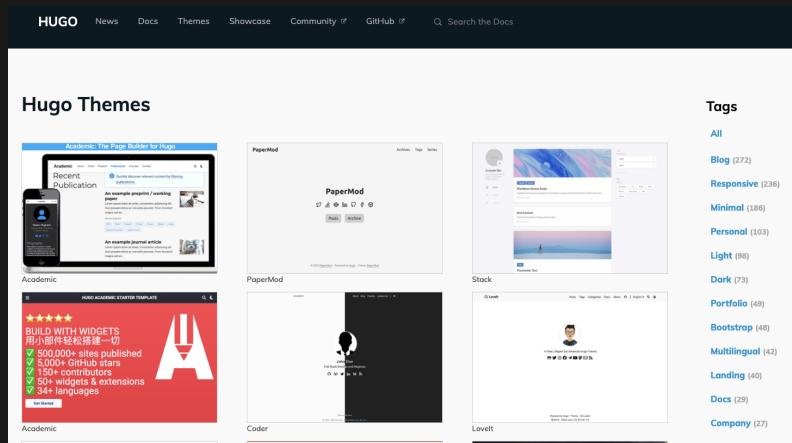
The screenshot shows a file explorer window with the following interface elements:

- Top bar: History, Files, Plots, Connections, Build, Viewer.
- Toolbar: New Folder, Delete, Rename, More.
- Breadcrumb navigation: Home > Reporting > Testfolder > testblogdown > test > themes > hugo-lithium.
- File list table with columns: Name, Size, Modified.
- Content of the file list:

| | Name | Size | Modified |
|-----|-------------|--------|------------------------|
| ... | .. | | |
| | .gitignore | 40 B | Apr 28, 2021, 12:07 AM |
| | archetypes | | |
| | exampleSite | | |
| | layouts | | |
| | LICENSE.md | 1.1 KB | Apr 28, 2021, 12:07 AM |
| | README.md | 2 KB | Apr 28, 2021, 12:07 AM |
| | static | | |
| | theme.toml | 482 B | Apr 28, 2021, 12:07 AM |

Choose your theme

See [here](#) for a collection of Hugo themes.



Recommended workflow: create a new site with desired theme

```
# create a new site with the anatole
# theme
blogdown::new_site(theme =
  "1xndrblz/anatole")
```

Existing site, new theme:

```
blogdown::install_theme(theme = "1xndrblz/anatole")
```

1. Move the configuration file (config.yaml or config.toml) from the themes/theme-name/exampleSite/ directory to the root directory of your website to match the newly installed theme.
2. Carefully inspect the differences between your new theme's exampleSite and the files inside your content/ folder. A theme's exampleSite content is tailored to a specific theme.

Archetypes

- They are like a template for the content on your site
- You may create your own archetypes! See the blog posts [here](#) and [here](#)

```
.  
└── categories.md  
└── default.md  
└── tags.md
```

Layouts

- They control the looks of elements on your site
- Since you should not touch the theme folder's files, this is where you can easily customize certain parts of your website
- See [here](#) for an example

```
<footer>
  {{ partial "foot_custom.html" . }}
  {{ with .Site.Params.footer }}
    <hr/>
    {{ . | markdownify }}
    {{ end }}
  </footer>
  </body>
</html>
```



Add new content



Want to change the landing page?

You need to do two things:

1. add an `_index.md` file to `content/` (not in a subfolder)
2. update the `index.html` layout



Setting global options in .Rprofile

```
library(usethis)
edit_r_profile()
options(blogdown.author = "Author Name",
       blogdown.ext = ".Rmd",
       blogdown.subdir = "blog",
       blogdown.yaml.empty = TRUE, # preserves empty fields in YAML header
       blogdown.new_bundle = TRUE, # creates index.md file (using bundles - storing files related to post in post folder)
       blogdown.title_case = TRUE)
```

[Global options](#)

[More global options](#)

Workflow

- Open your website project, click the "Serve Site" addin
- Revise old pages/posts, or click the "New Post" addin
- Write and save (take a look at the automatic preview)
- Push everything to Github

Blogdown resources

- [Blogdown demo site](#)
- [Blogdown book](#)
- [Blog post](#)
- [Blogdown workshop](#)
- [Yihui's slides from RStudio Conf](#)

Your turn!



1. Create a new blogdown project with the theme of your choice (follow the workflow described [here](#))
2. Connect your blogdown project to a remote GitHub repository
3. Create a netlify account and connect netlify with GitHub
4. Publish your blogdown page through netlify

Other useful R tools ★

Xaringan



See [here](#) for more info.

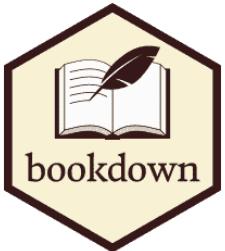
Bookdown

BOOKDOWN

Write HTML, PDF, eBook, and Kindle books with R Markdown

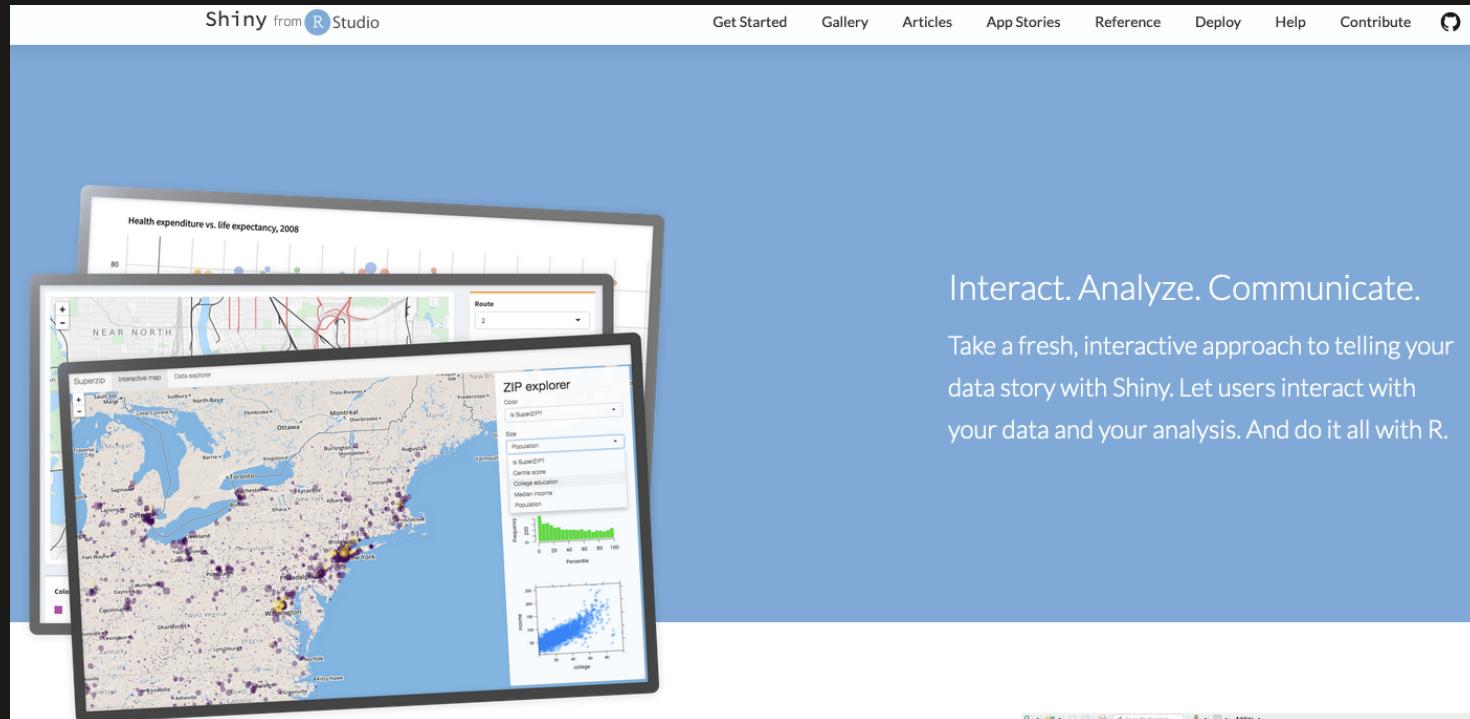
The `bookdown` package is an [open-source R package](#) that facilitates writing books and long-form articles/reports with R Markdown. Features include:

- Generate printer-ready books and eBooks from R Markdown documents.
- A markup language easier to learn than LaTeX, and to write elements such as section headers, lists, quotes, figures, tables, and citations.
- Multiple choices of output formats: PDF, LaTeX, HTML, EPUB, and Word.
- Possibility of including dynamic graphics and interactive applications (HTML widgets and Shiny apps).
- Support a wide range of languages: R, C/C++, Python, Fortran, Julia, Shell scripts, and SQL, etc.
- LaTeX equations, theorems, and proofs work for all output formats.
- Can be published to GitHub, bookdown.org, and any web servers.
- Integrated with the RStudio IDE.
- One-click publishing to <https://bookdown.org>.



See [here](#) for more info.

Shiny applications



Interact. Analyze. Communicate.

Take a fresh, interactive approach to telling your data story with Shiny. Let users interact with your data and your analysis. And do it all with R.

See [here](#) for more info