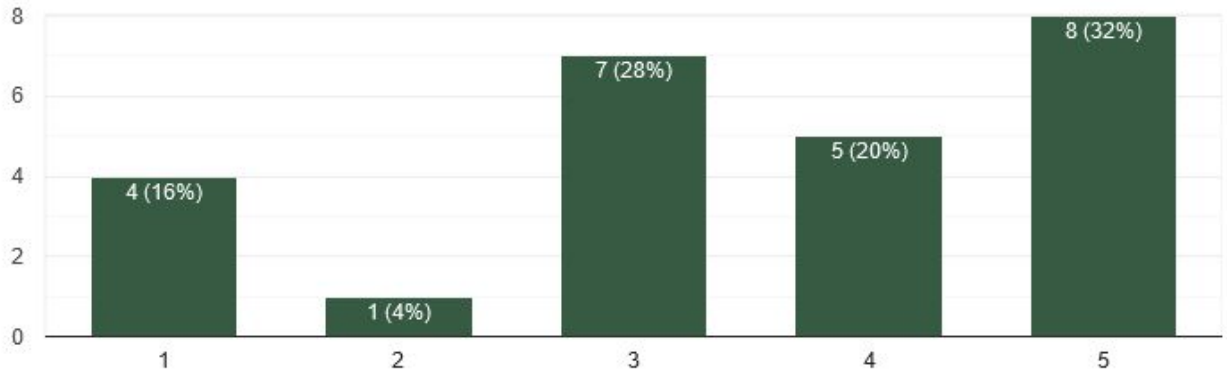
Project-based workflows with GitHub

Courtney Robichaud and Emma Hudgins
@cdrobich @emmajhudgins

You walk away confident in using Git/GitHub for version control with your (R-based) projects

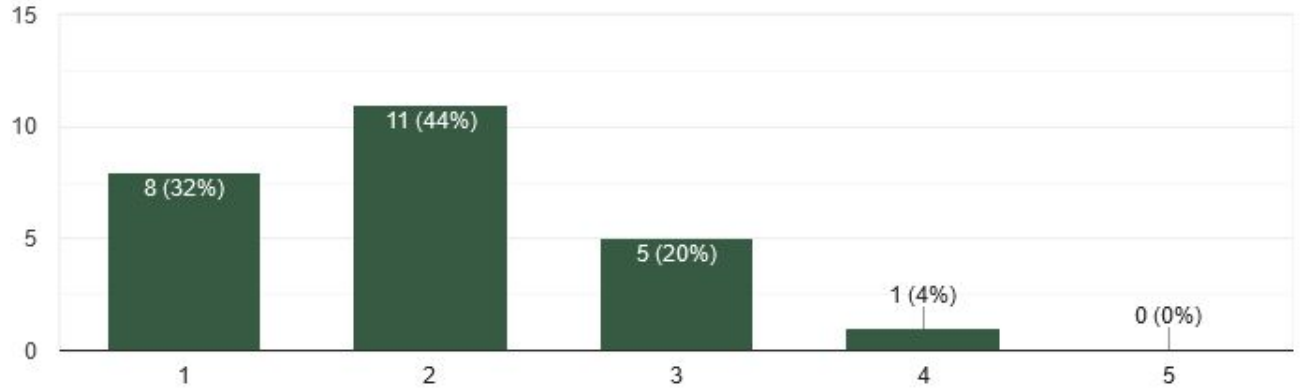
What's your level of familiarity with R and RStudio?

25 responses



What is your level of familiarity with Git/GitHub?

25 responses



What are your concerns about using or learning Git/GitHub?

Little coding experience

Never got the flow of it

The learning curve/difficulty for new users

I don’t know how to use it in my research

What is your main priority when it comes to developing GitHub skills?

25 responses

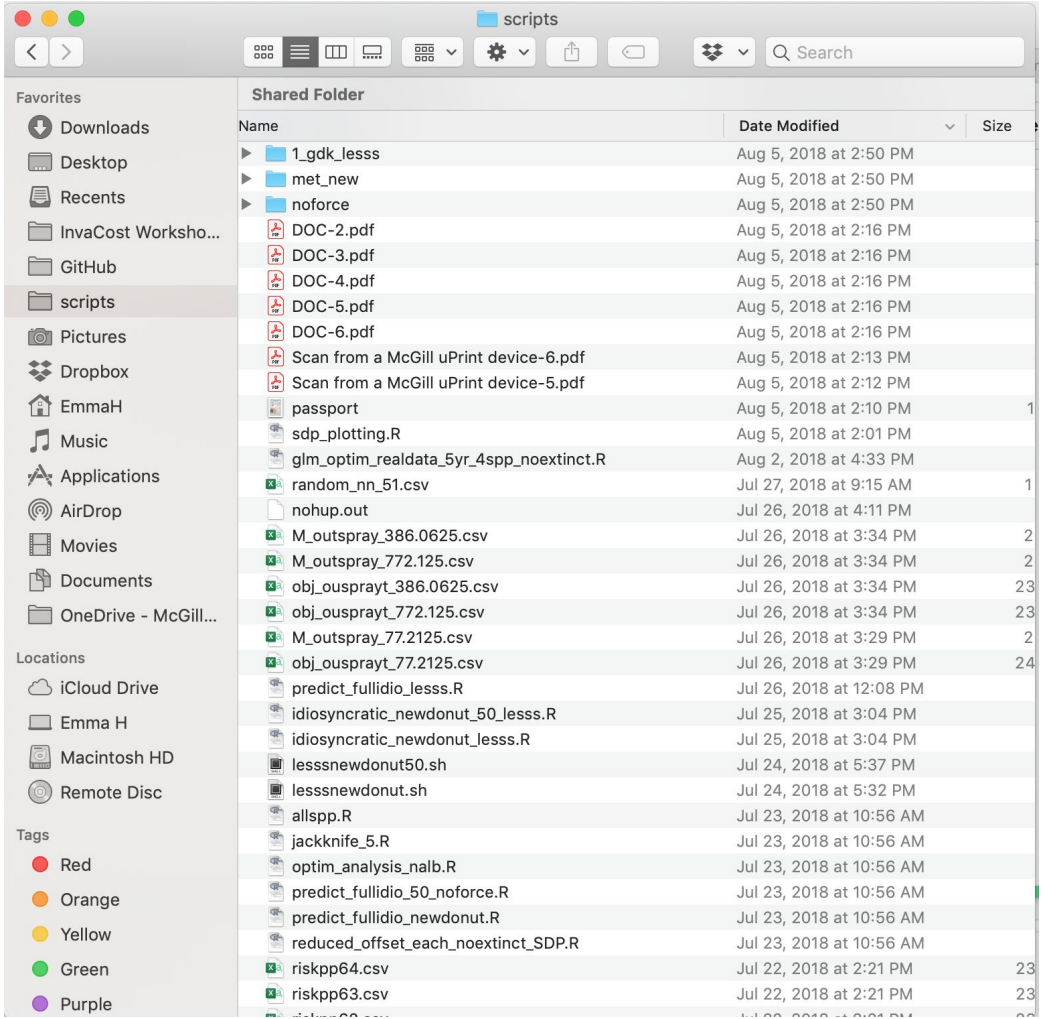
What we will cover:

- What are R projects, Git, and Github?
- Walking through how to use them
- Demo creating a repo, writing script, generating figures, and committing them to a repo
- You follow along with our demo and make your own!

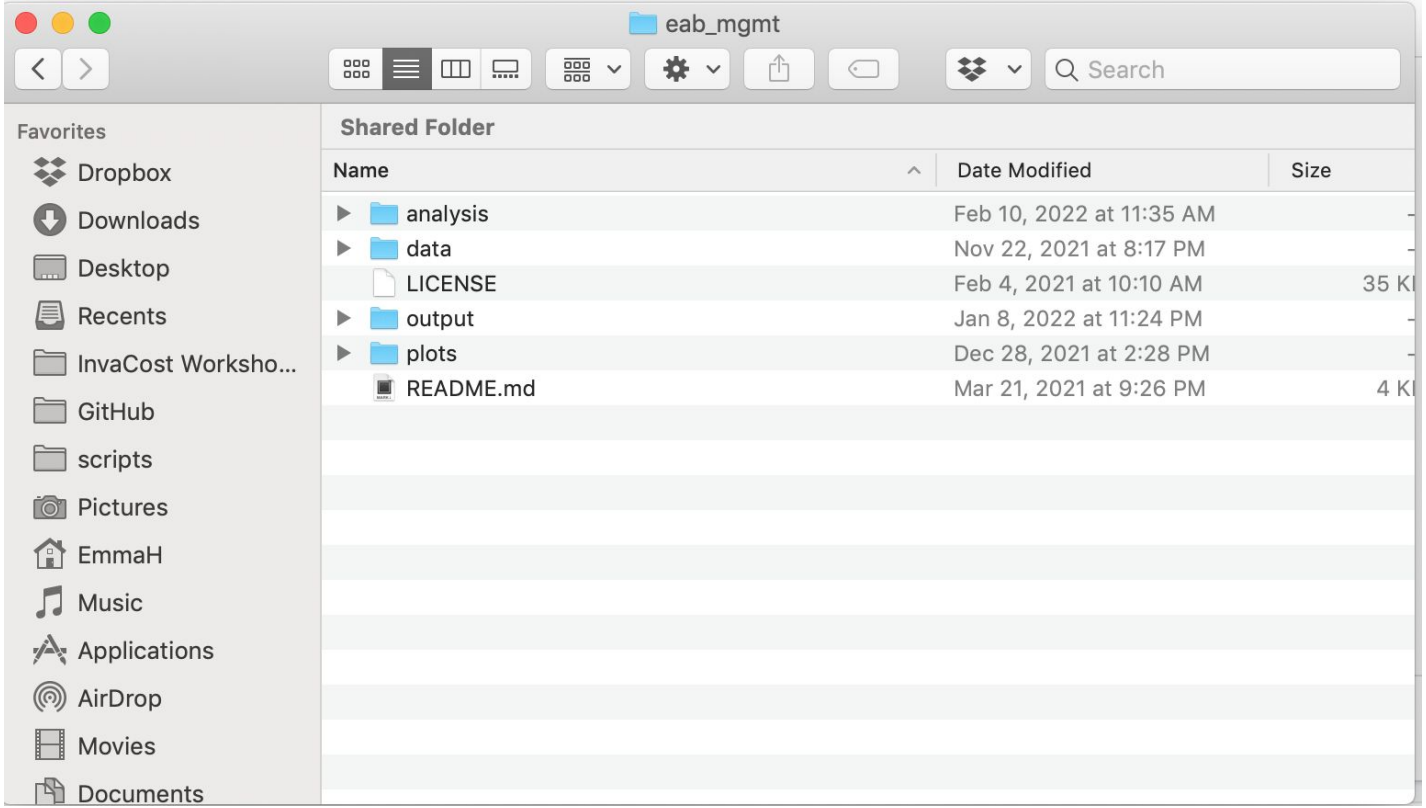
The power of projects, Git and GitHub

Your current organization

Could look something like this



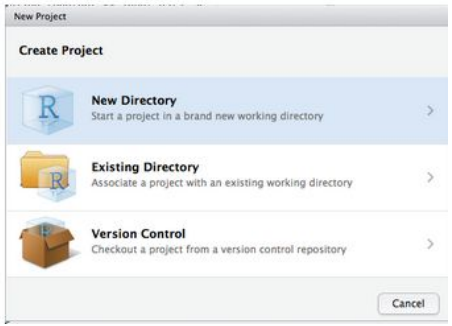
Your ideal organization



Push to GitHub

Commit often

R project



R Projects

```
setwd("C:/Users/cdrobich/Desktop/M.Sc/Birds")
```

code and data

Thursday, January 13, 2022 at 1:44 PM

nestinitdata_final.csv
77 KB

TRIG PR path linkag...
5.3 KB

Download All • Preview All

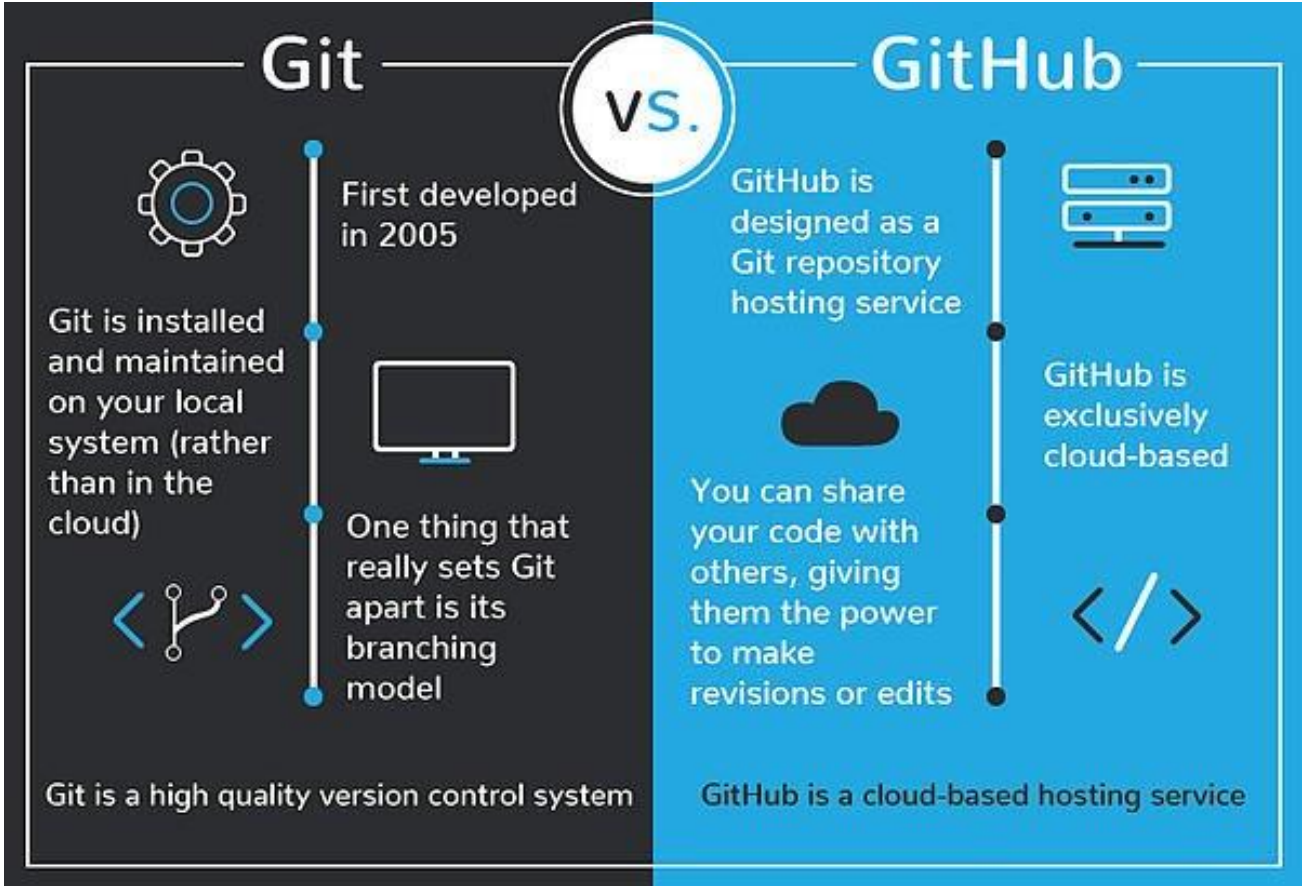
You replied to this message.

[External Email]

Hi Emma,
Thanks again for your help today! The code and data are attached if you're able to help with the code for that plot.
Thanks!

What is a project and why is it better?

- All your data (raw and manipulated), scripts, and output are saved in separate folders
- Does not call on anything system-specific
- Well commented so others (including future you) understand


Git
VS.
GitHub
First developed in 2005
Git is installed and maintained on your local system (rather than in the cloud)
One thing that really sets Git apart is its branching model
GitHub is designed as a Git repository hosting service
GitHub is exclusively cloud-based
You can share your code with others, giving them the power to make revisions or edits
Git is a high quality version control system
GitHub is a cloud-based hosting service

What else can GitHub do?

- Can be a **cloud storage** service for any type of file
- “**Forking**” allows people to use others’ projects as **templates** for their own
- Provides a **hosting** service for web content
- Allows you to freeze your work at a given moment in time as a **‘release’** which can be linked to a DOI (Required by many journals/funders)
- Provides **integration** with other tools (e.g. OSF)

GitHub basics

Push, pull, commit, fork, clone

Create your project in GitHub and ‘clone’ to your machine. Then you will interact with Git by:

Repo(sitory) - one or more folders that have git functionality, GitHub repos are stored on the cloud

Push - send changes to the cloud

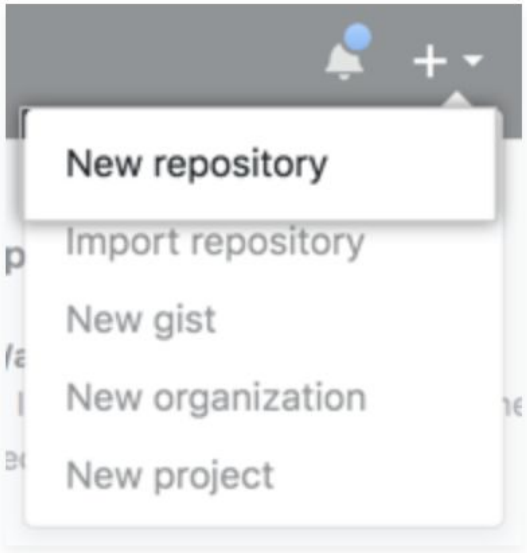
Pull - get changes from the cloud

Commit - create a named version of a set of one or more changes to the repo

Clone - copy an existing repo into your local github folder **such that it communicates with the original repo**

Fork - freeze an existing repo in time and copy it into your github folder **such that it does not communicate with the old repo**

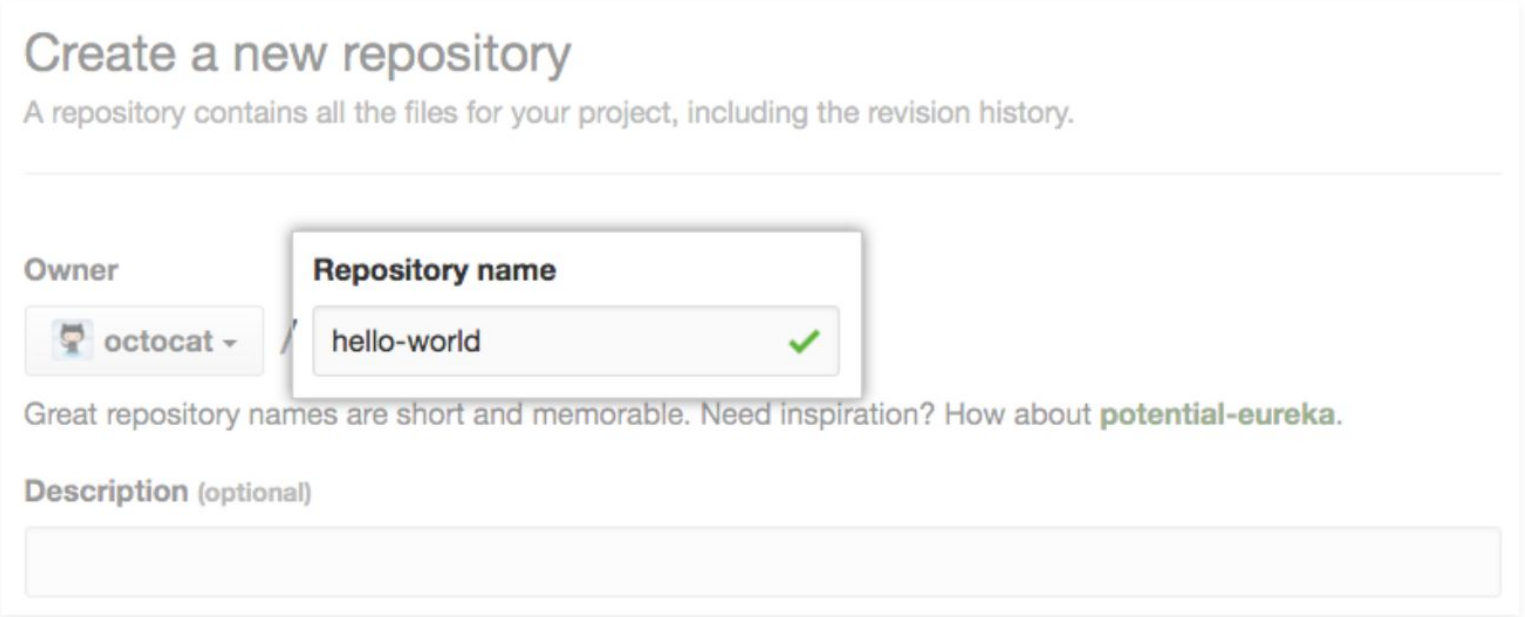
1. In the upper-right corner of any page, use the + drop-down menu, and select **New repository**.



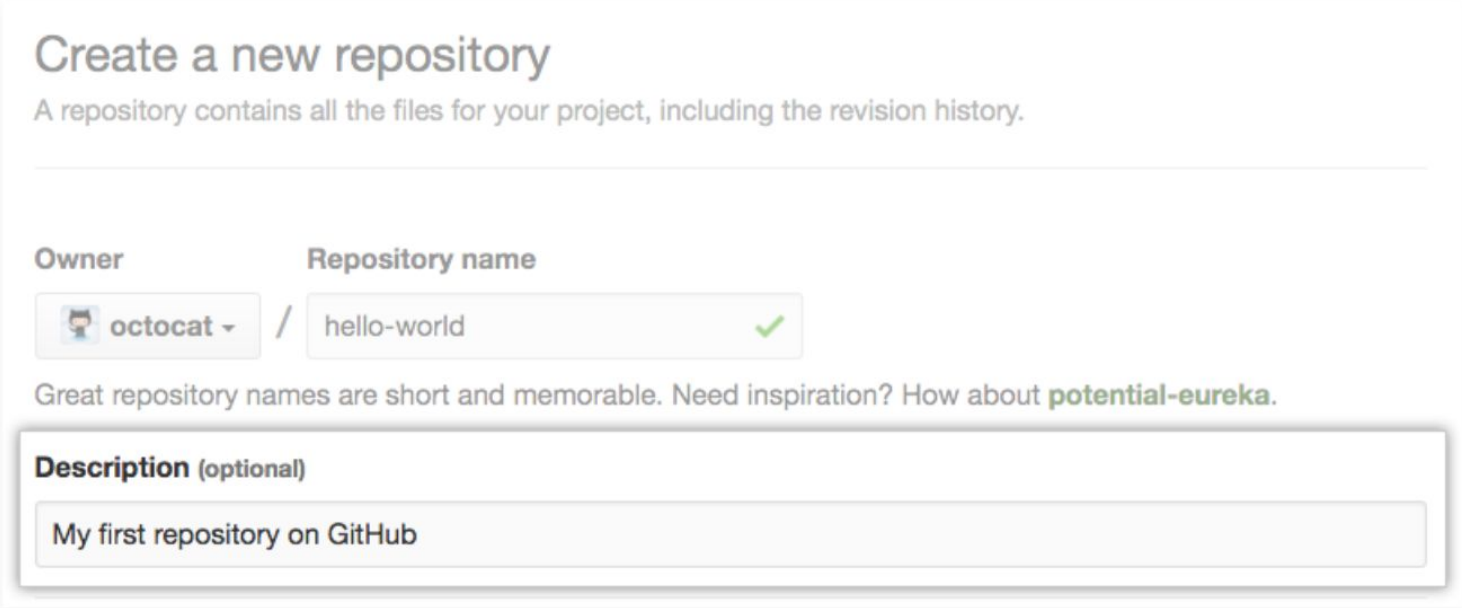
OR click the green button in the left pane

github.com

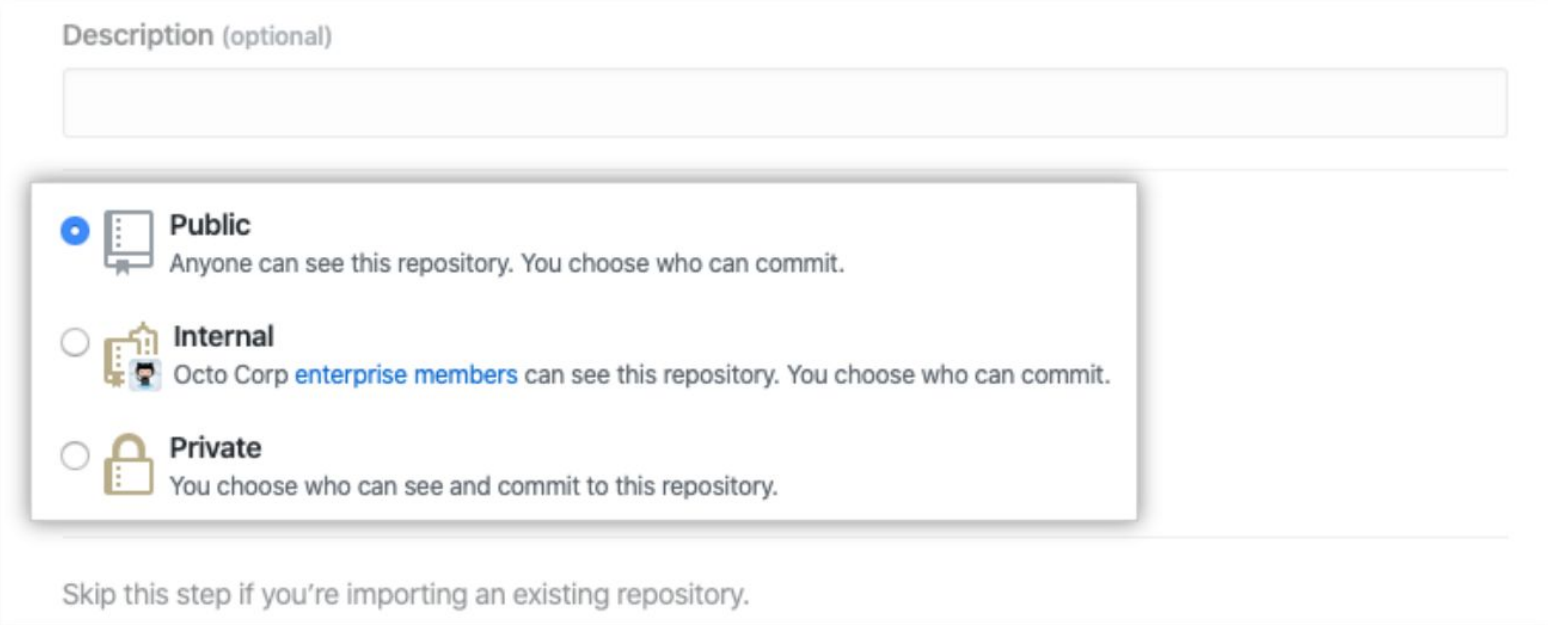
2. Type a short, memorable name for your repository. For example, "hello-world".



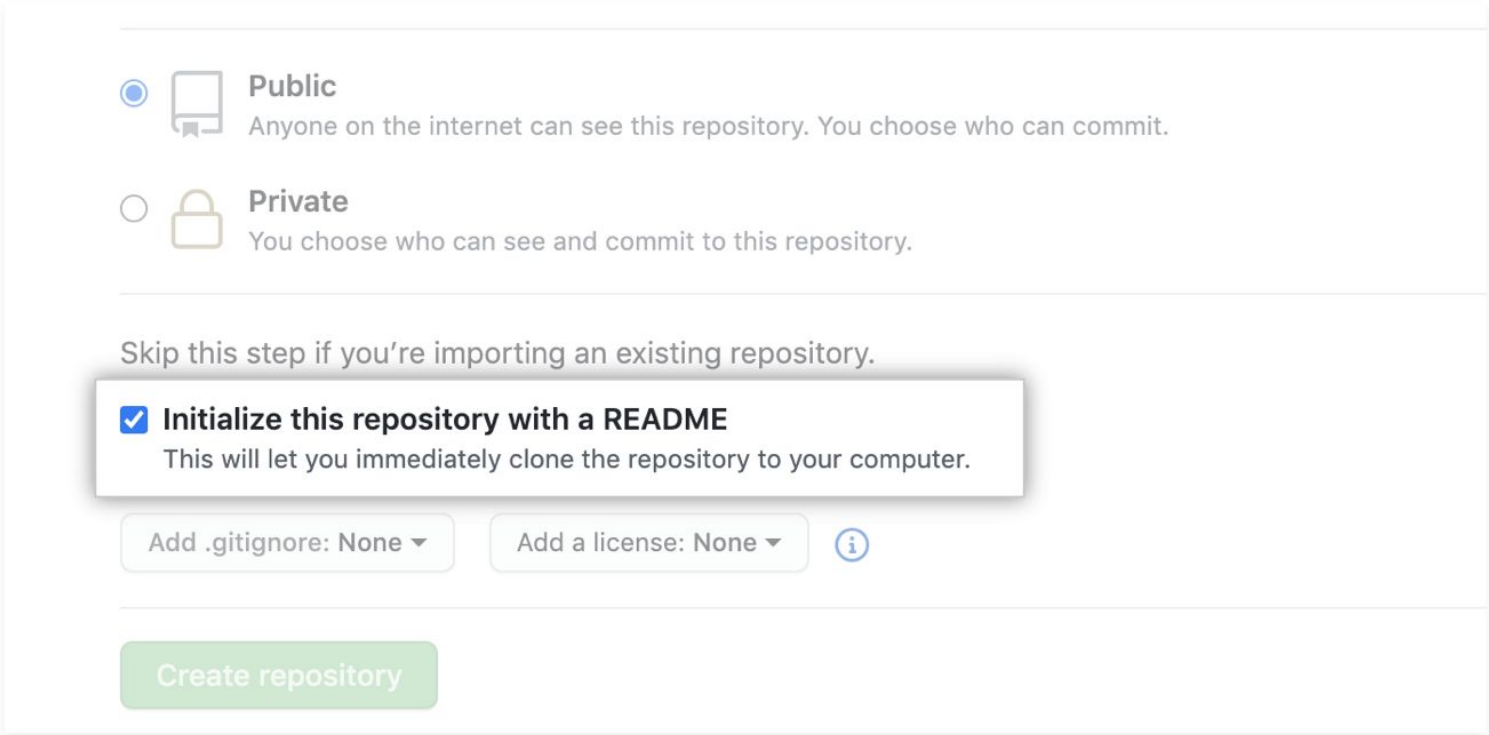
3. Optionally, add a description of your repository. For example, "My first repository on GitHub."



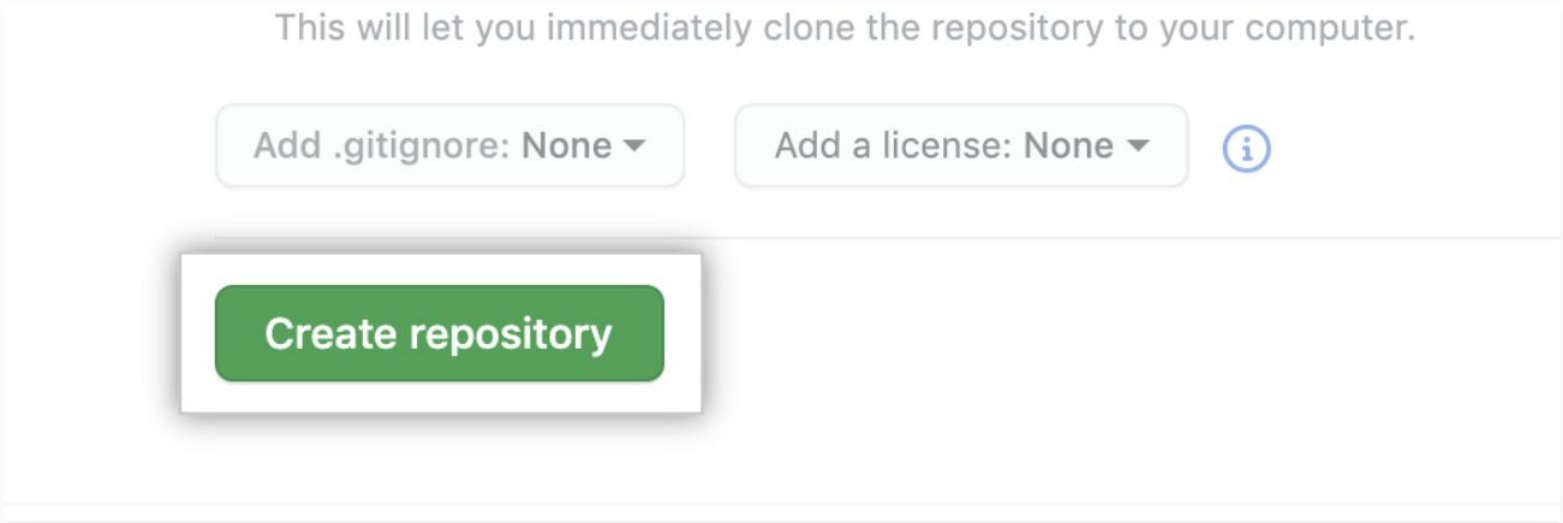
4. Choose a repository visibility. For more information, see "About repositories."



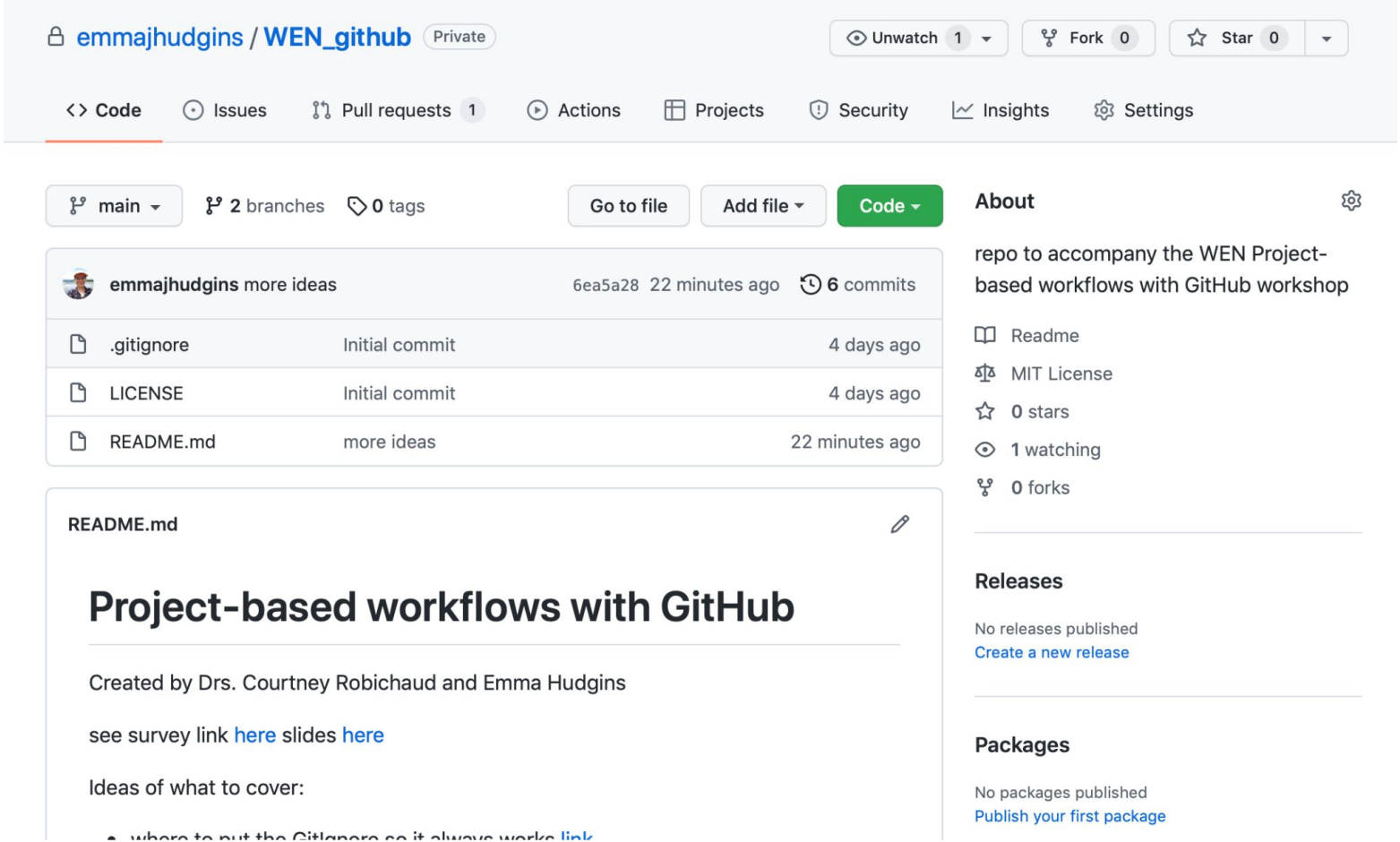
5 Select **Initialize this repository with a README.**

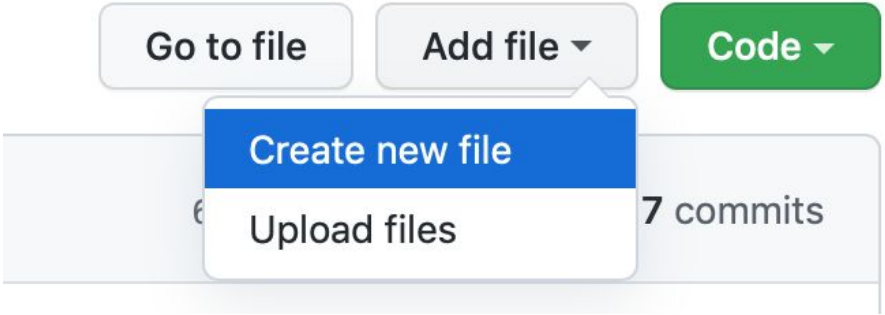


6 Click **Create repository**.



Structure of a repo





Commit new file

Create README.md

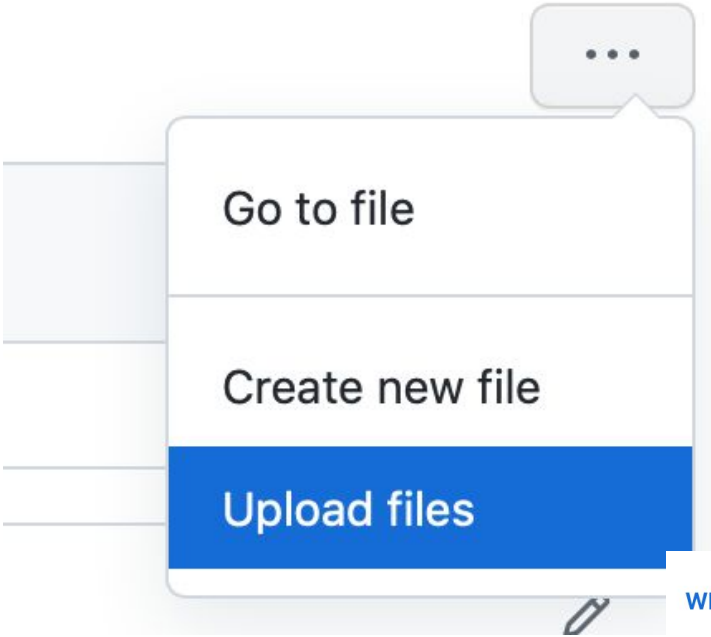
Add an optional extended description...

Commit directly to the `main` branch.

Create a **new branch** for this commit and start a pull request. Learn more about pull requests.

Commit new file

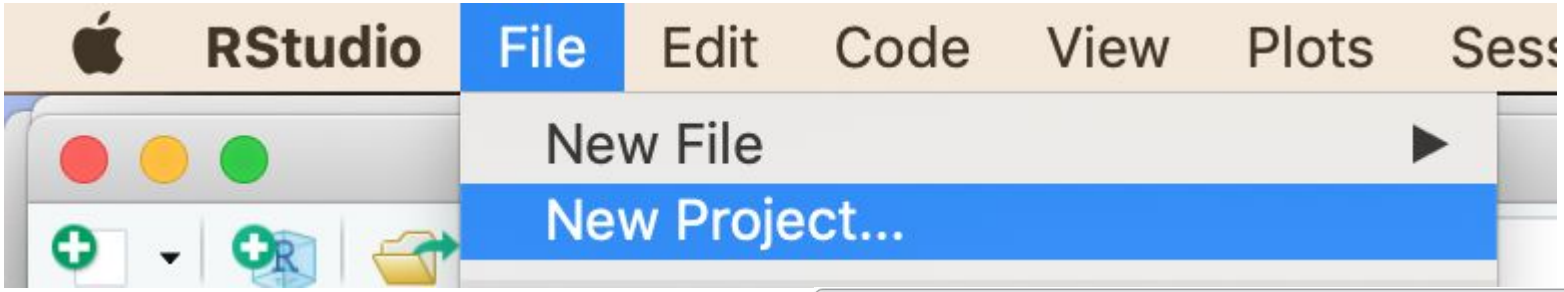
Cancel

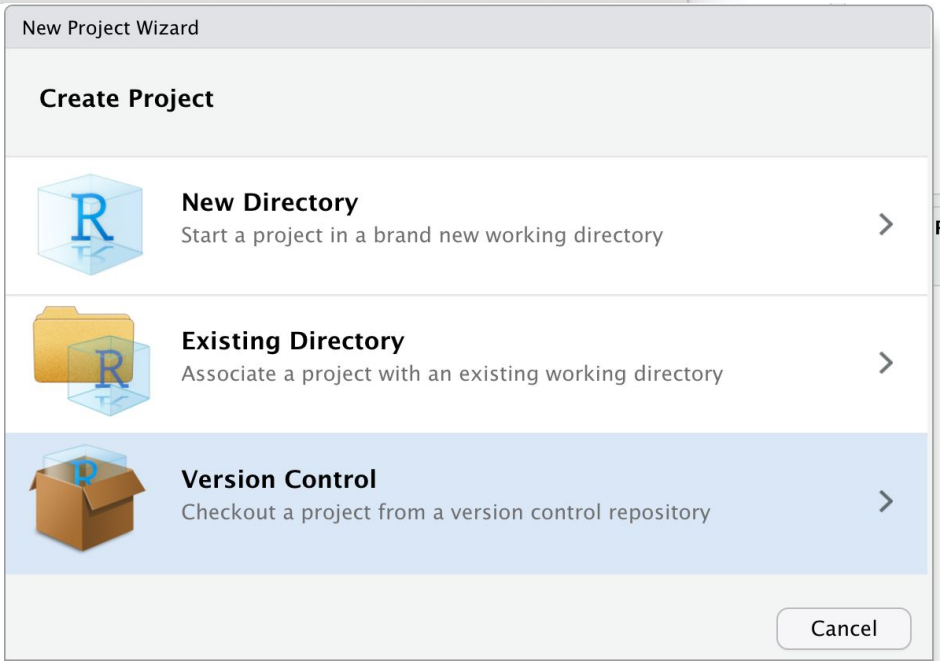


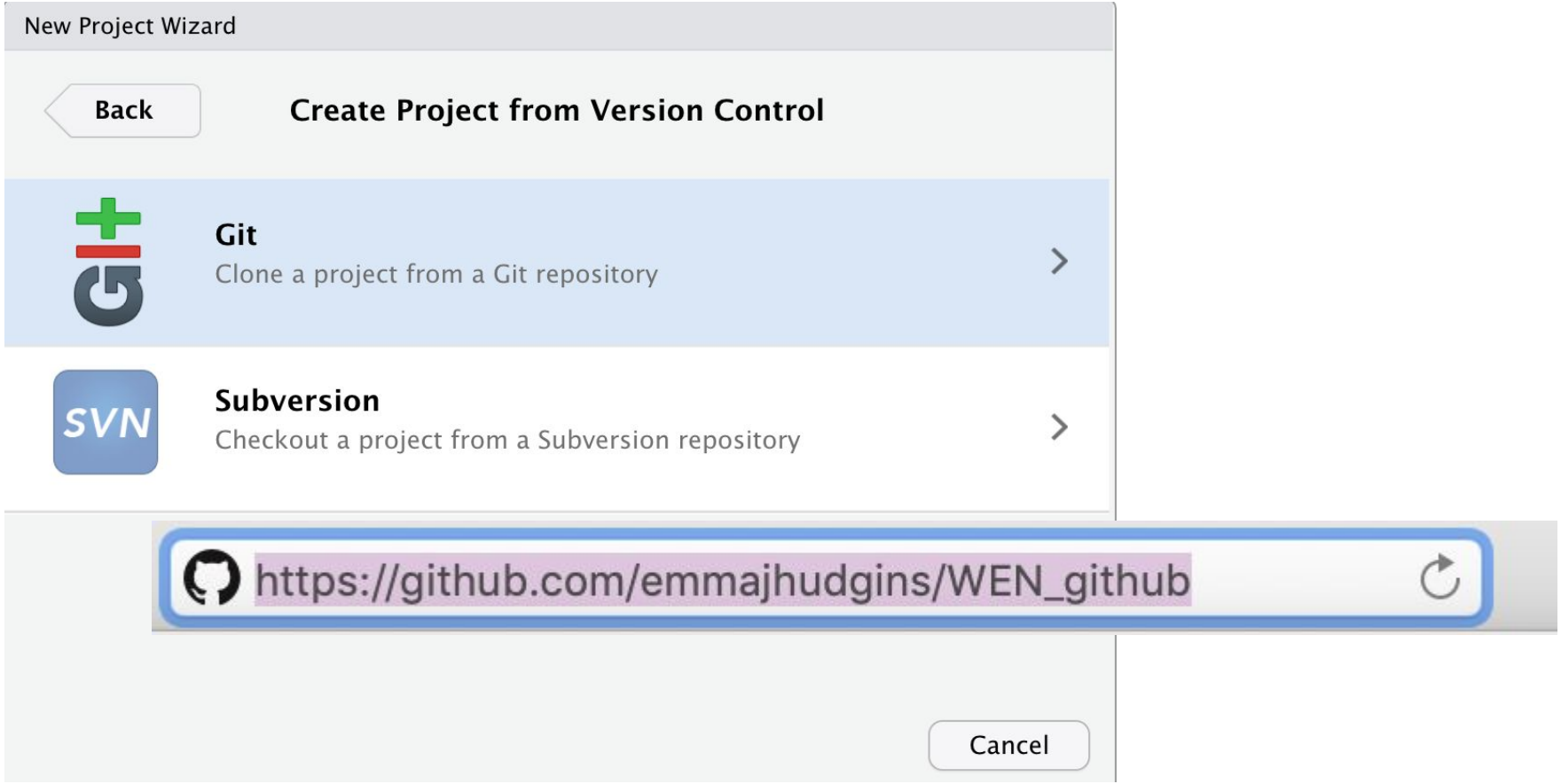
WEN_github /

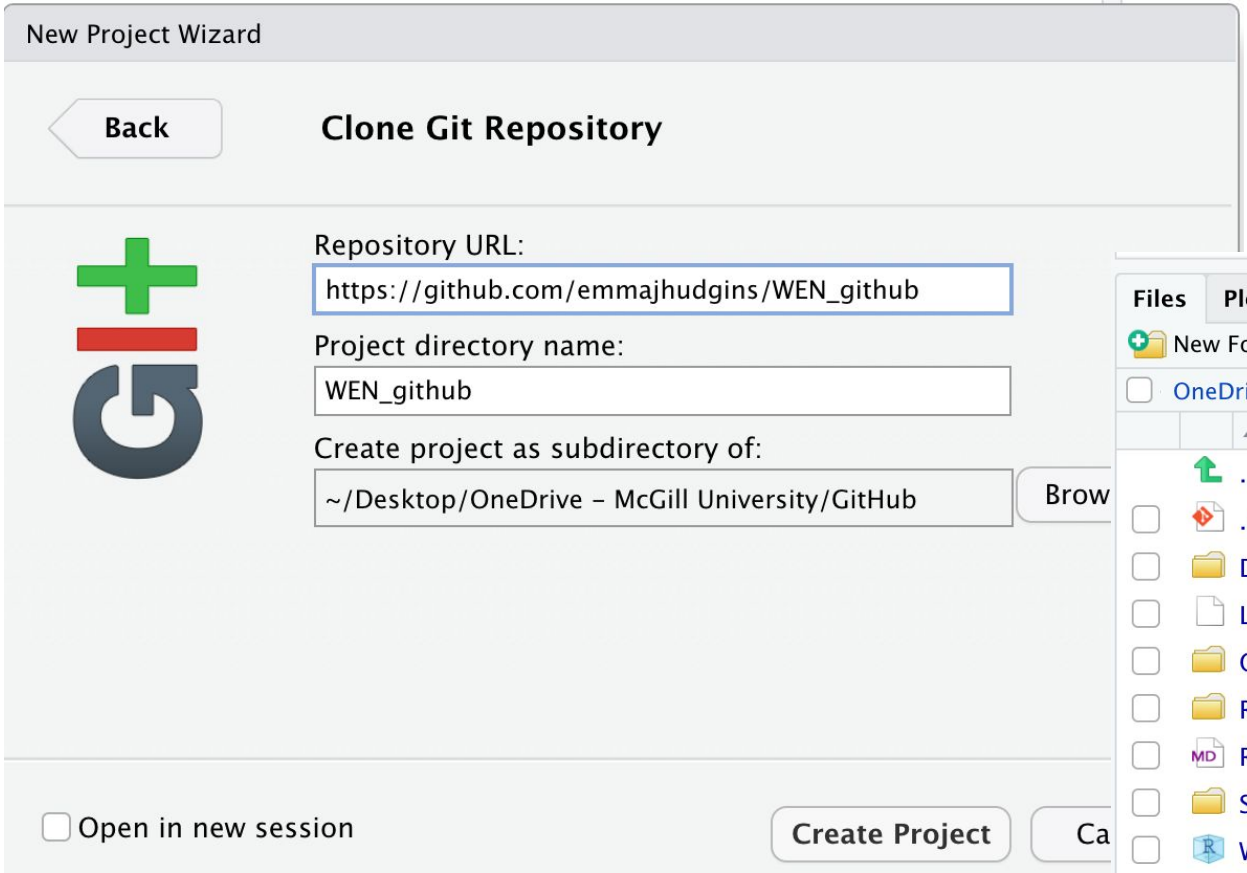
Drag files here to add them to your repository

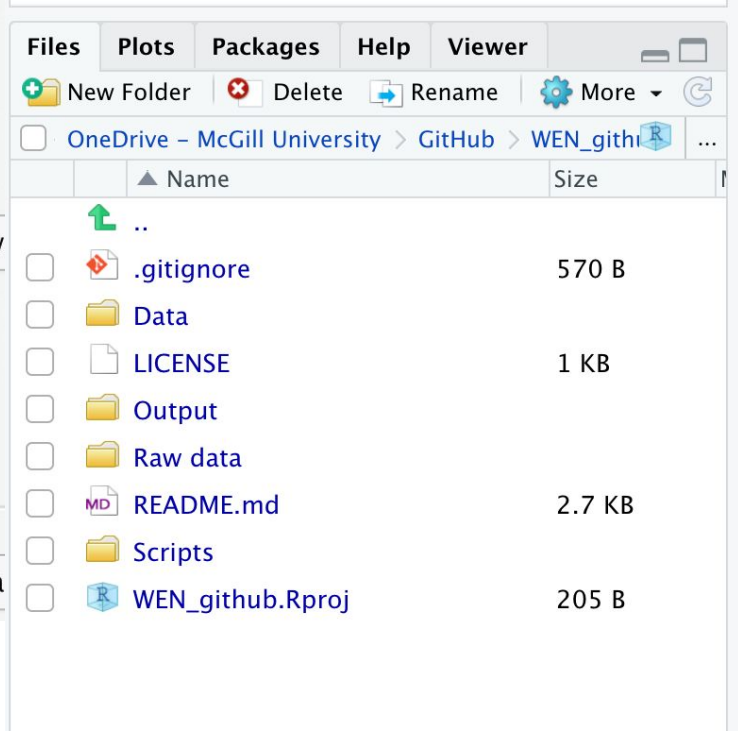
Or choose your files


RStudio
File
Edit
Code
View
Plots
Sess
New File
New Project...

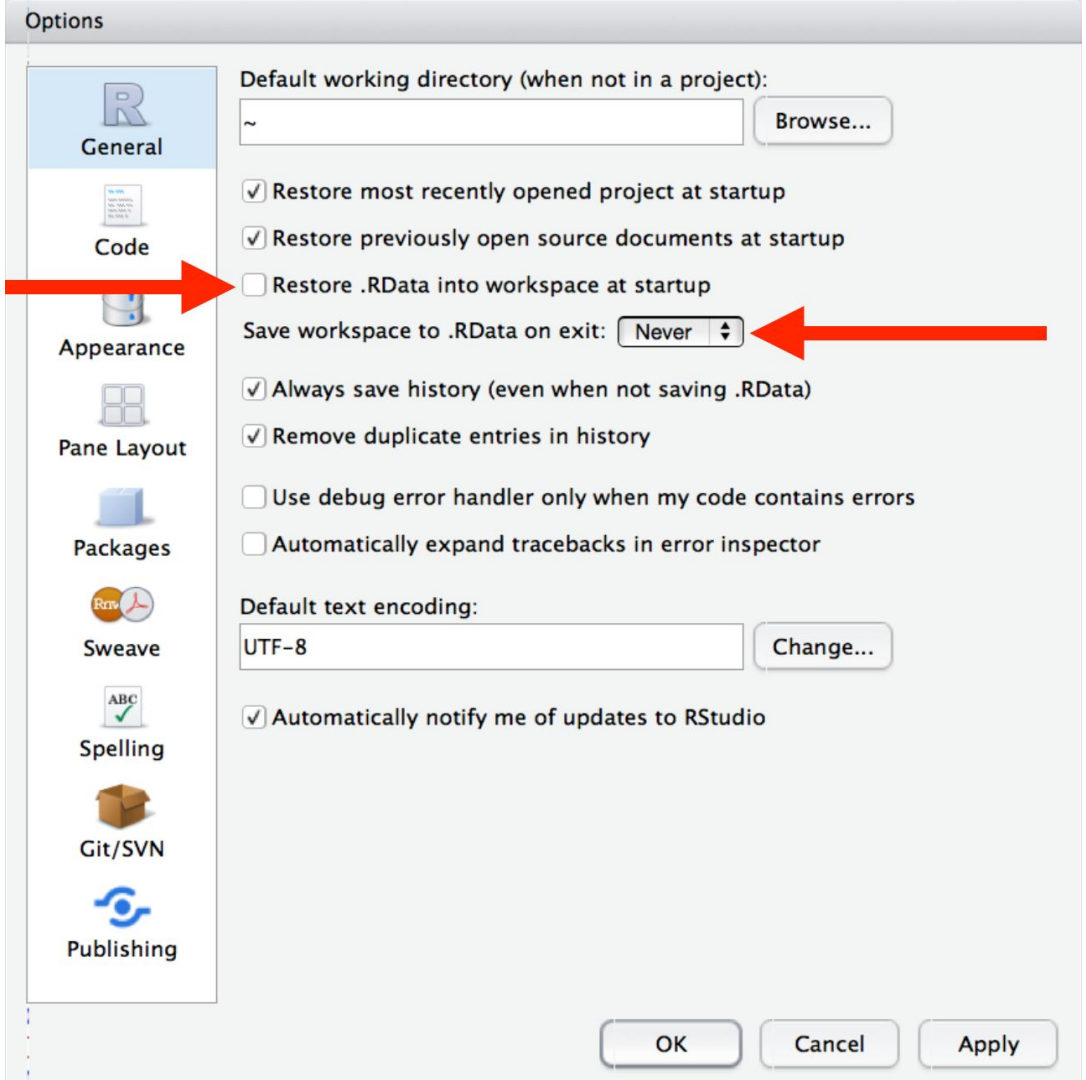

New Project Wizard
Create Project
New Directory
Start a project in a brand new working directory
Existing Directory
Associate a project with an existing working directory
Version Control
Checkout a project from a version control repository
Cancel


New Project Wizard
Back
Create Project from Version Control
Git
Clone a project from a Git repository
SVN
Subversion
Checkout a project from a Subversion repository
https://github.com/emmajhudgins/WEN_github
Cancel


New Project Wizard
Back
Clone Git Repository
Repository URL:
https://github.com/emmajhudgins/WEN_github
Project directory name:
WEN_github
Create project as subdirectory of:
~/Desktop/OneDrive - McGill University/GitHub
Brow
Open in new session
Create Project
Ca


Files
Plots
Packages
Help
Viewer
New Folder
Delete
Rename
More
OneDrive - McGill University
GitHub
WEN_githu
Name
Size
..
.gitignore
570 B
Data
LICENSE
1 KB
Output
Raw data
README.md
2.7 KB
Scripts
WEN_github.Rproj
205 B

Check/change your settings in R:



GO TO GITHUB

.gitignore

Choose a template based on your main programming language (R template ignores files like `.RHistory`)

Some examples of files you probably want to ignore:

- Sensitive information (e.g. passwords)
- Binary files such as `.Rdata`.
- Files > 50MB. Git is specifically made for **code** (e.g. `.R`) and does not intend to track all changes in large data files (these can be uploaded in ‘releases’ with DOIs through Zenodo.
- *temporary files/folders* with ‘disposable’ content

Choosing the best license

I need to work in a community.

Use the **license preferred by the community** you're contributing to or depending on. Your project will fit right in.

If you have a dependency that doesn't have a license, ask its maintainers to **add a license**.

I want it simple and permissive.

The **MIT License** is short and to the point. It lets people do almost anything they want with your project, like making and distributing closed source versions.

Babel, **.NET Core**, and **Rails** use the MIT License.

I care about sharing improvements.

The **GNU GPLv3** also lets people do almost anything they want with your project, *except* distributing closed source versions.

Ansible, **Bash**, and **GIMP** use the GNU GPLv3.

choosealicense.com

Ideal folder structure

Raw Data

Metadata includes date of download or collection, original source and re-use info

(Derived) Data

Data you transformed after downloading/collecting, e.g. merging 2 databases

Scripts

Code (can separate by language)

Output

Figures, tables, results

Every folder should contain a README!

Readme/Metadata best practices

- Include package version information and any external software used
- Describe files in a logical order
- Describe any column/variable names (especially units)

File naming

- Be as descriptive as possible
- Can add leading numbers to scripts that indicate order they should be run e.g.
- 01-data_processing.R
- 02-model_fitting.R
- Avoid dates/overly generic names
- Name output similarly to script that generated it
- Use hyphens and dashes

Clean coding

Be proactive

- Use #### #### to separate steps
- Describe each major step and why it’s done
- Put yourself in the shoes of the person reading the code for the first time
- Include code author names, software versions

More advanced GitHub

More advanced functionality

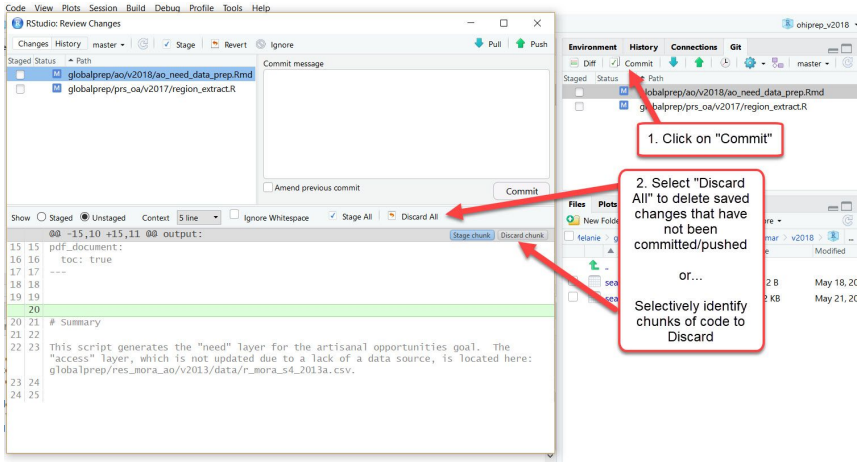
Branch - one set of version histories for a repo, including the '**main**' original branch, and additional branches used to suggest changes, test out new ideas that may not work etc.

Pull request - a suggested commit (created in another branch or from a fork) that must be approved by the owner of the main branch

Pull often, commit after each change

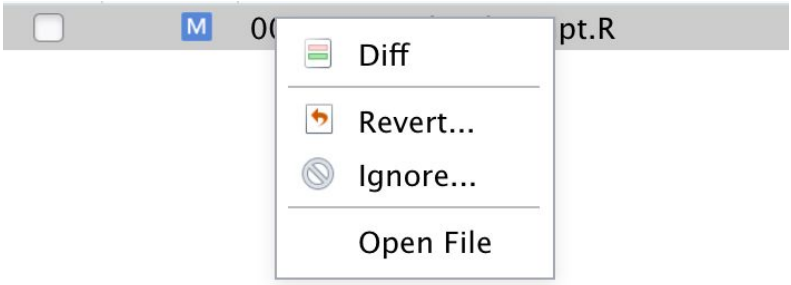
Revert changes

Easier pre-commit, but possible post-commit too.



Pre-commit:

In RStudio, right click on a file and select ‘revert’

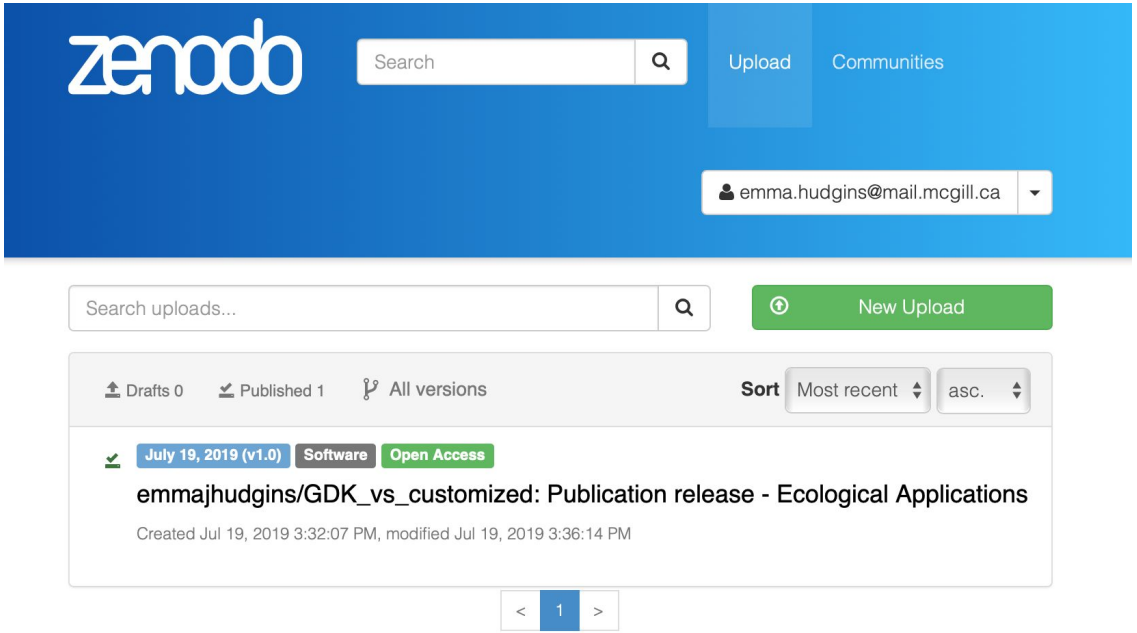


Releases, Zenodo & DOI creation

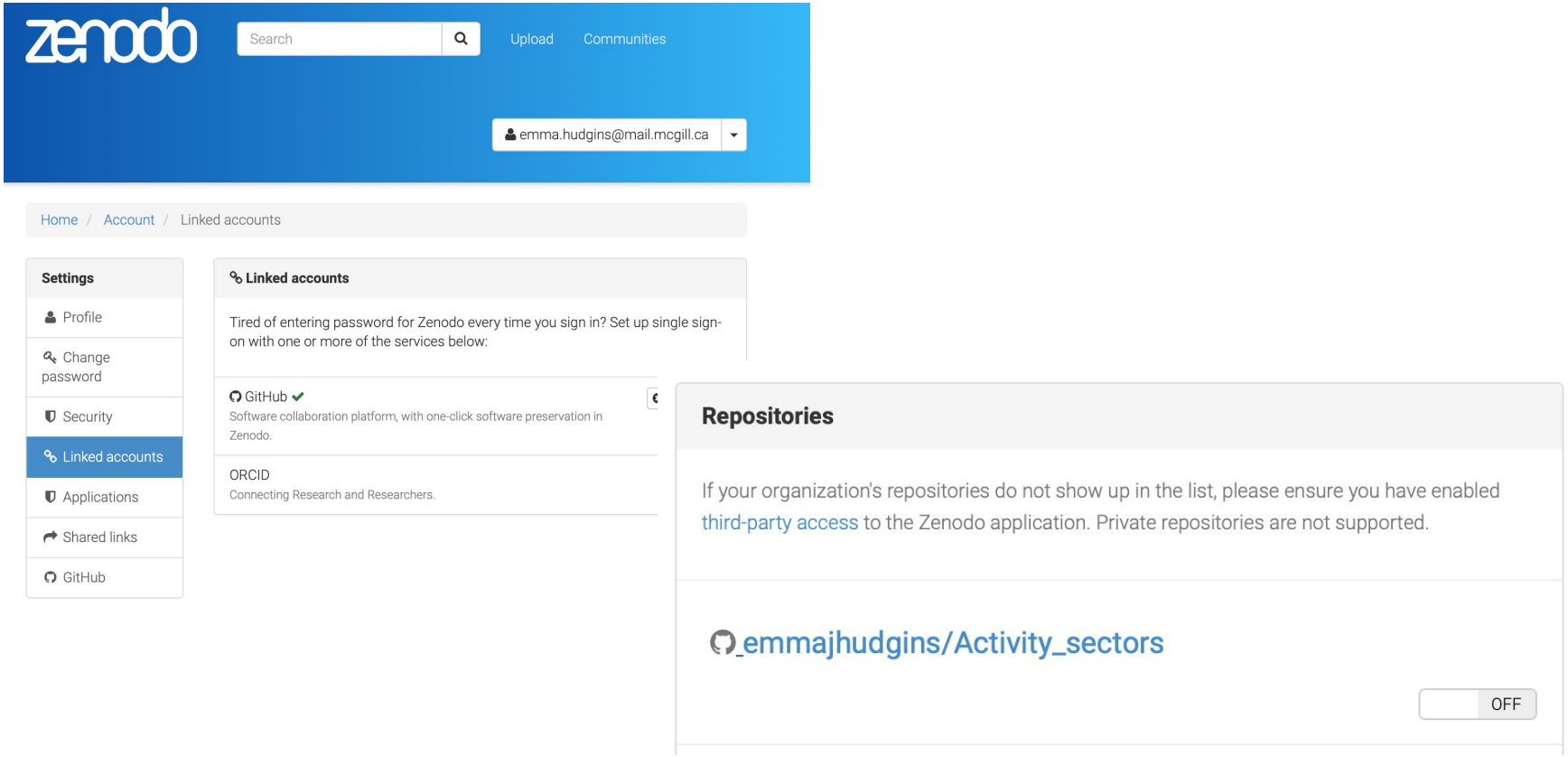
Releases

No releases published

Create a new release



Releases, Zenodo & DOI creation



OpenRefine

https://openrefine.org/

Other helpful resources

https://datacarpentry.org/rr-version-control

https://carpentries-incubator.github.io/git-Rstudio-course/

https://www.markdownguide.org/basic-syntax/