

Version Control

with GitHub Desktop

DR KIM MARTIN
29 AUG 2022

LinkedIn: kcmartin

Version Control with GitHub Desktop © Copyright 2022, Dr Kim Martin

This work was inspired by '**GitHub without the command line**' © Copyright 2020, CodeRefinery contributors.
<https://coderefinery.github.io/github-without-command-line/>

These instructional materials are made available under the [Creative Commons Attribution license](#). The following is a human-readable summary of (and not a substitute for) the [full legal text of the CC BY 4.0 license](#).

You are free:

- to **Share**—copy and redistribute the material in any medium or format
 - to **Adapt**—remix, transform, and build upon the material
- for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

- Attribution**—You must give appropriate credit (mentioning that your work is derived from work that is Copyright © Kim Martin and, where practical, linking to the source url), provide a [link to the license](#), and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

No additional restrictions—You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits. With the understanding that:

Notices:

- You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation.
- No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material.

Preparation:

- Create a GitHub account:
 - <https://www.github.com>
- Download and install GitHub Desktop:
 - <https://desktop.github.com/>
- Download and install *both* R and RStudio:
 - <https://posit.co/download/rstudio-desktop/>
- Download the 'Exercise files':
 - <https://github.com/RSE-at-SUN/GitHub-Desktop-workshop-2022>

... we are ALL here to help each other!



Why Git?

- Powerful
 - “Version control on steroids”
 - Can be used for large projects involving many people - e.g. Linux
- Popular
 - Used by many (most?) professional coders
 - Used extensively in Open Source / Open Science work
- Practical
 - Help you do your work efficiently, and work efficiently with others
 - Good skill for the CV!

Git



GitHub

```
Rterm (64-bit)
kcmartin@bosbou504824 MINGW64 ~
$ R
R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

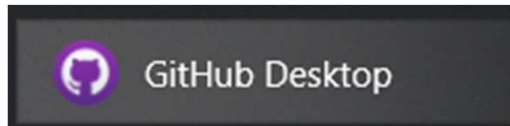
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

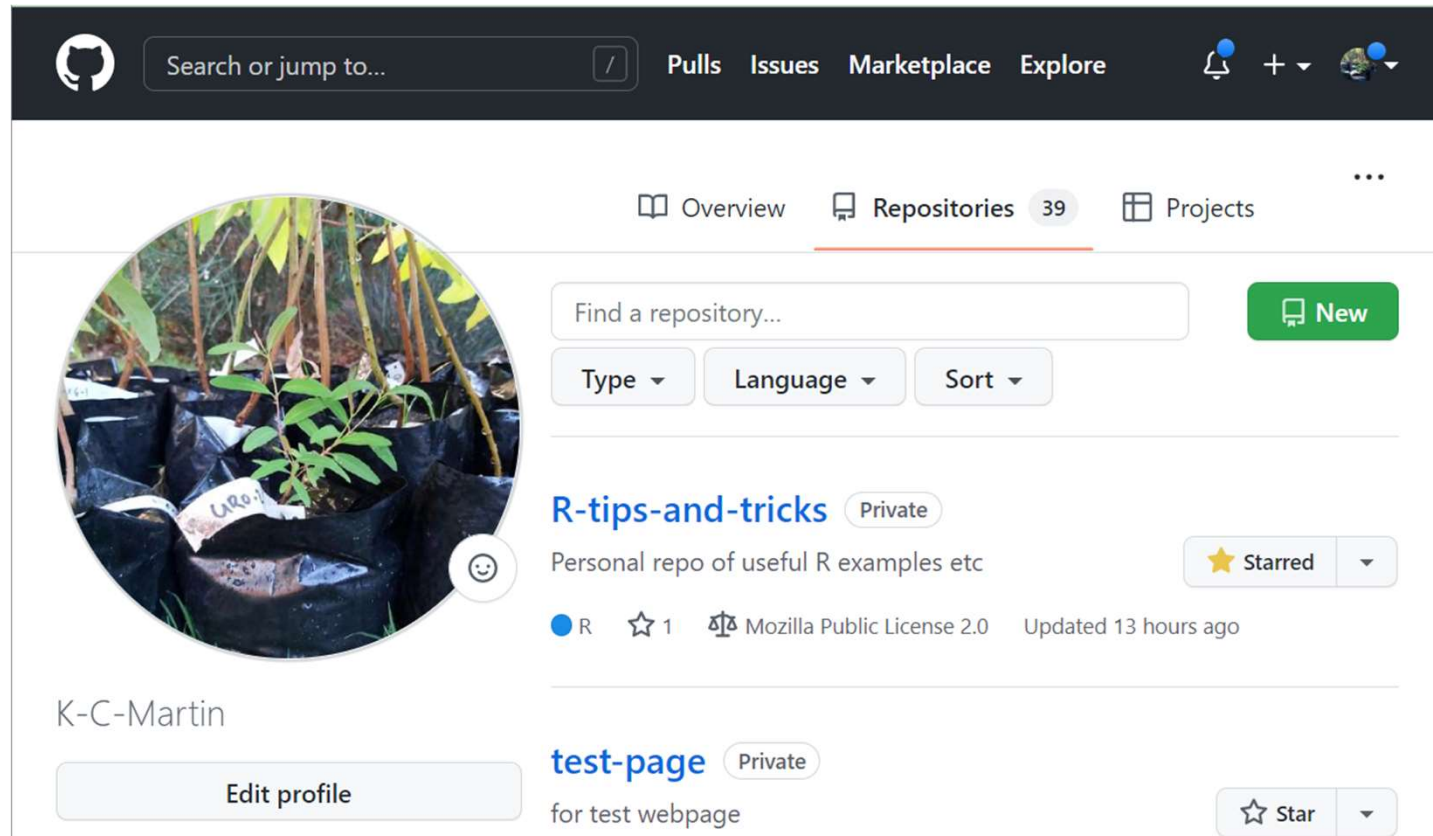
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

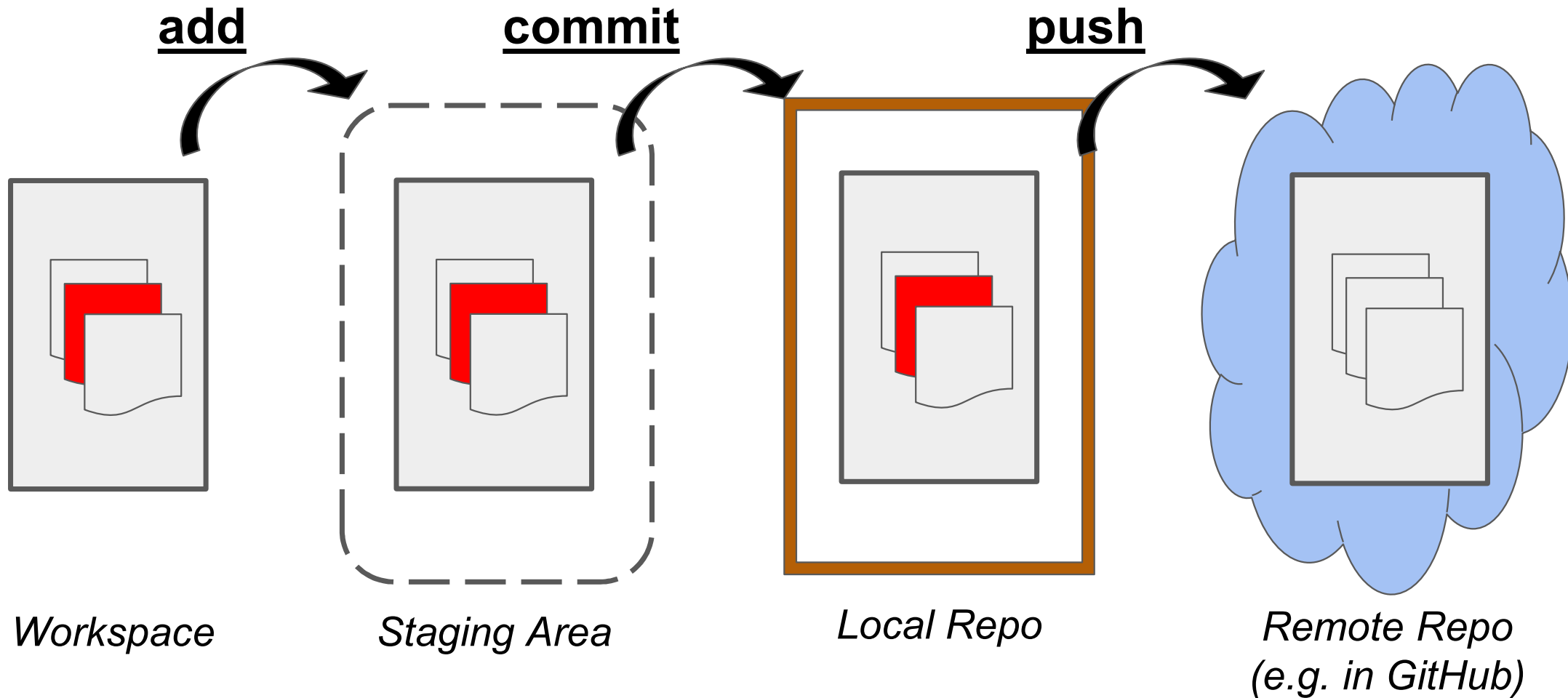


A version control software...
controlled by command-line and
some Graphical User Interfaces
(GUIs) ... e.g. 'GitHub Desktop'

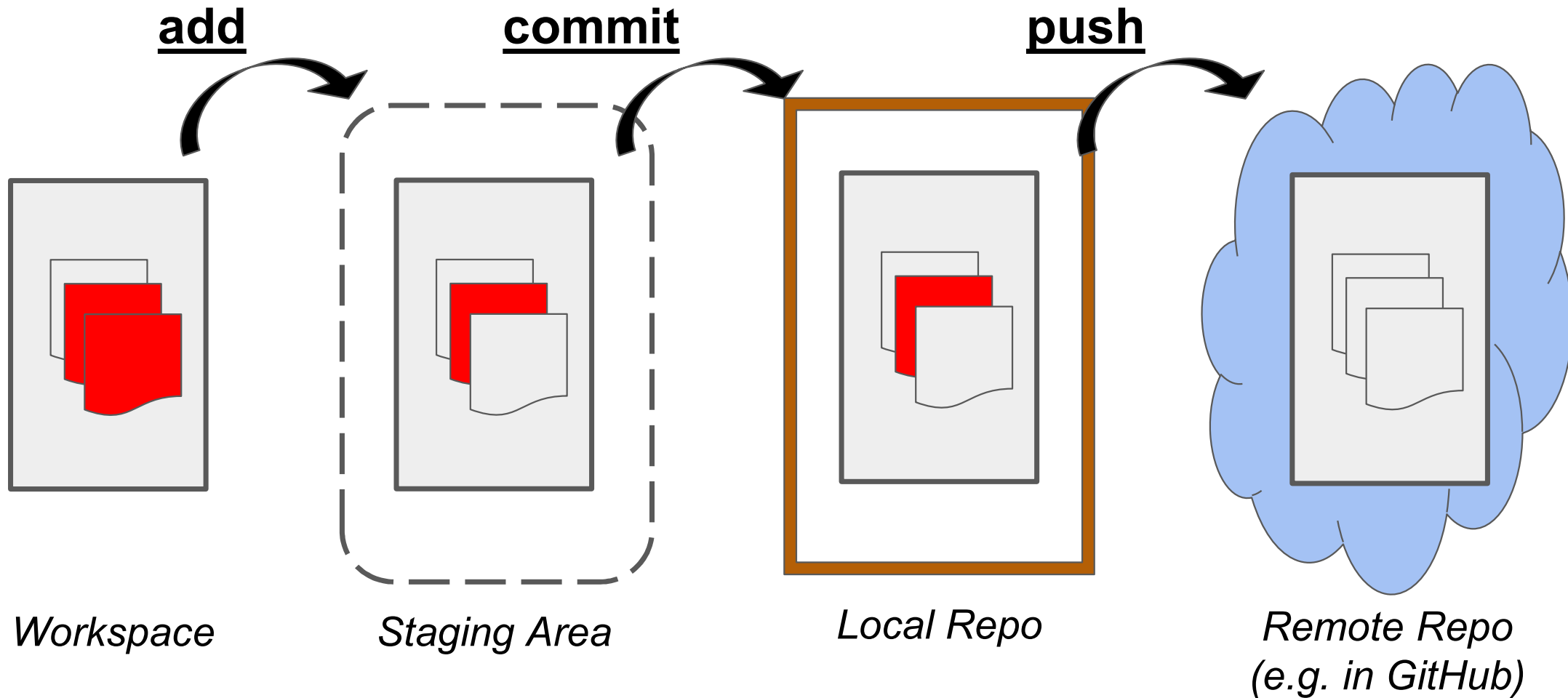


An online (cloud) host to backup and share git repositories (repos)

Basic idea - staging and committing

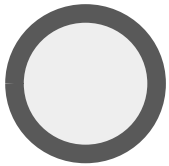


Basic idea - staging and committing



Commit history - 'snapshot' of each new commit (with ID)

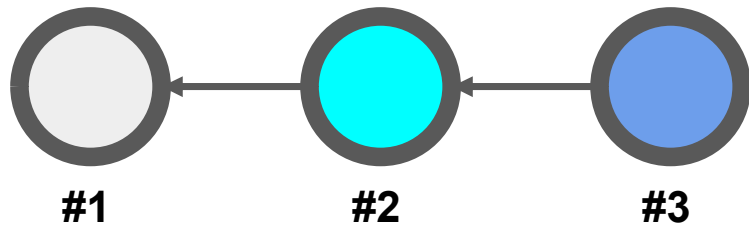
INSIDE REPO:



#1

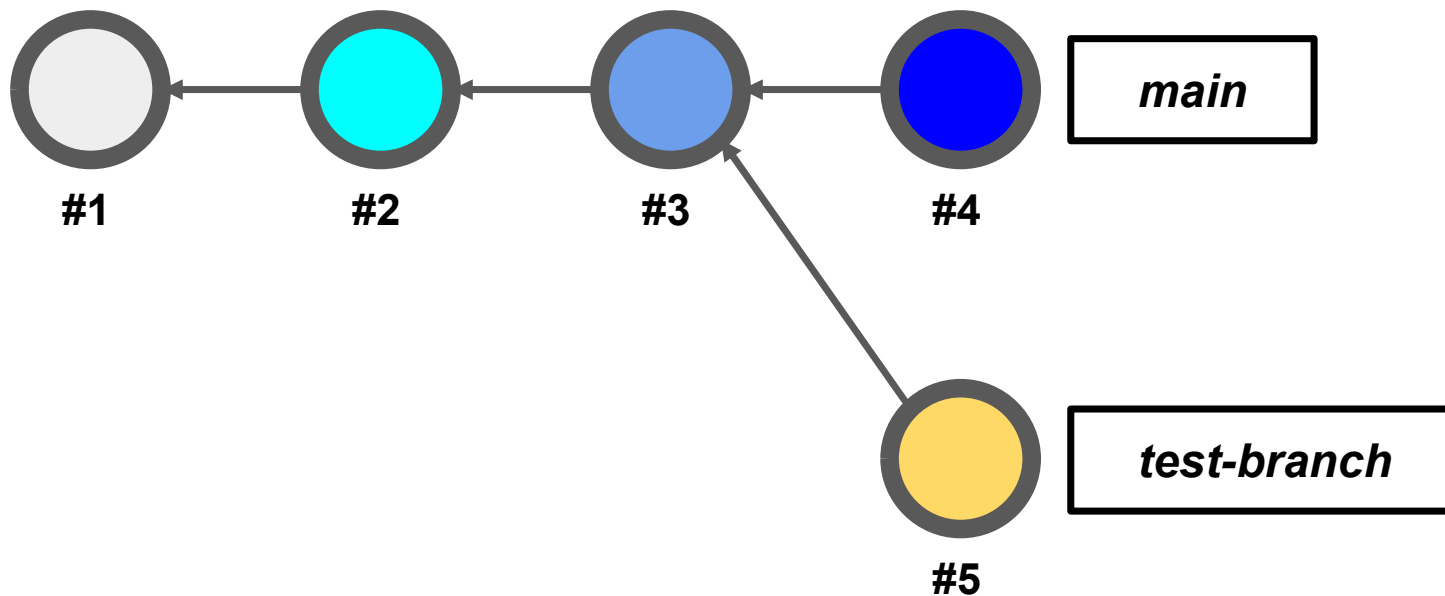
Commit history - timeline of version changes (with IDs)

INSIDE REPO:



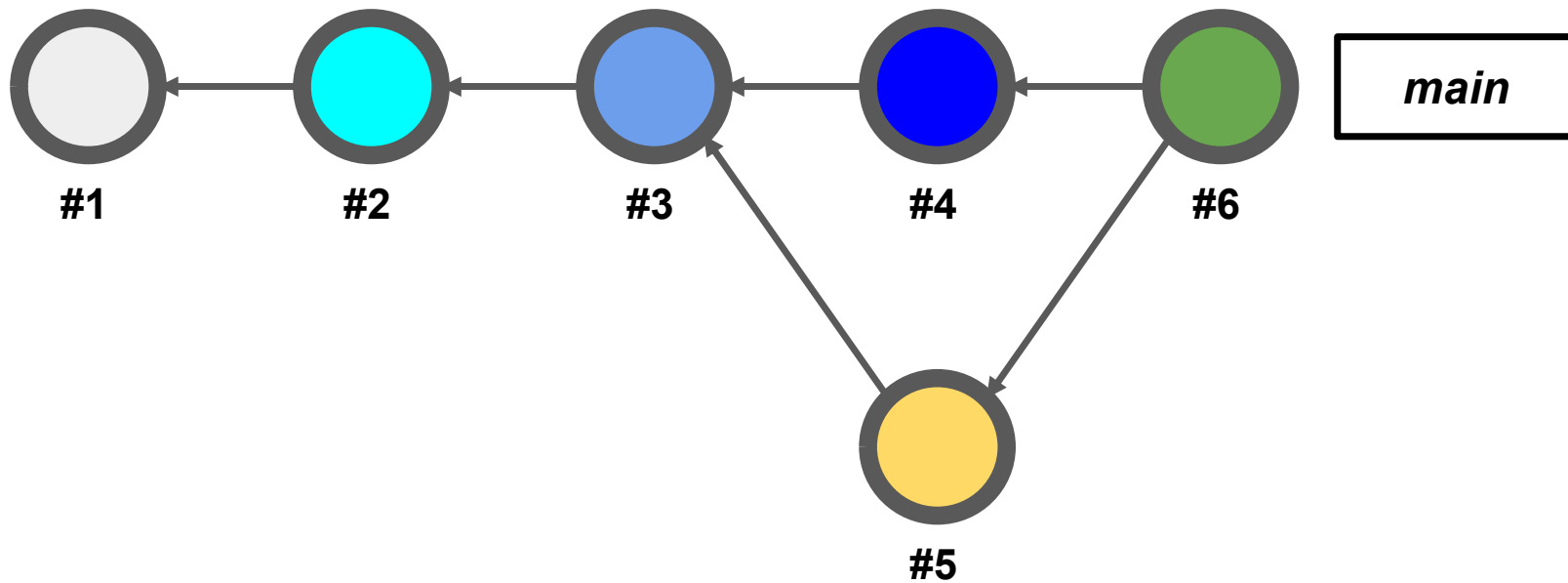
Commit history - branches (trying out different edits)

INSIDE REPO:



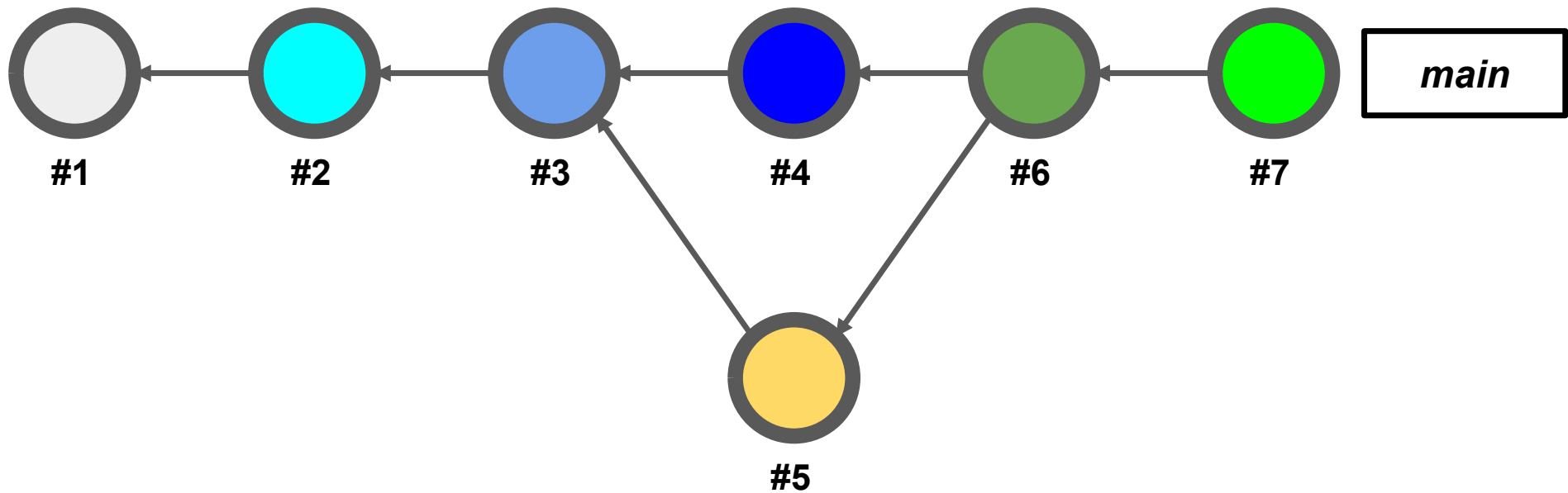
Commit history - branches can be merged

INSIDE REPO:



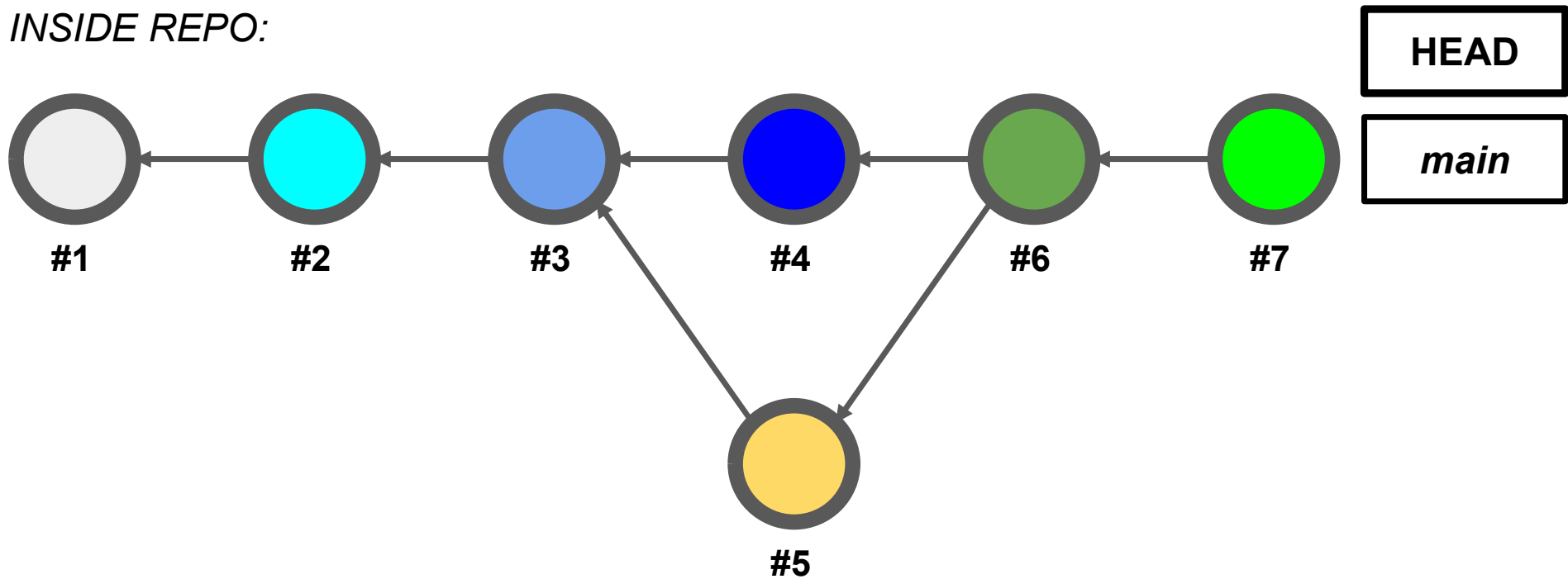
Commit history - branch names refer to most recent commit

INSIDE REPO:



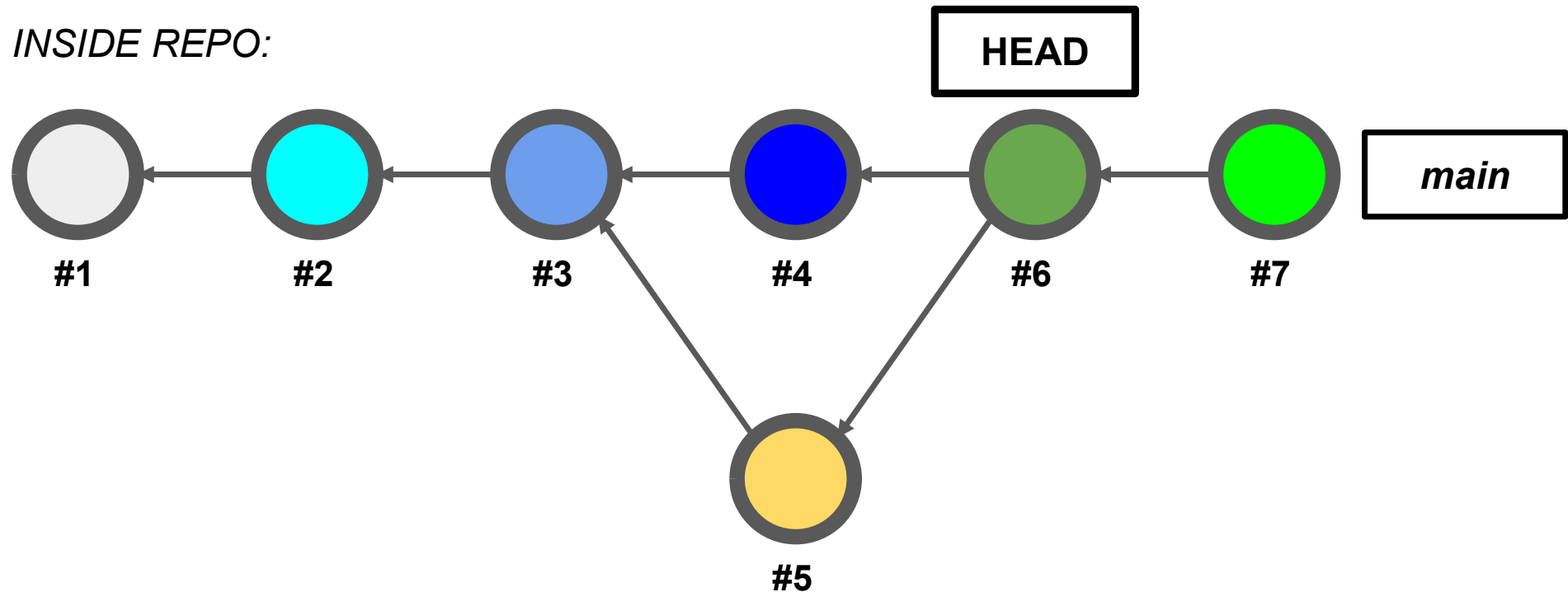
Commit history - HEAD is the current state of the repo

INSIDE REPO:



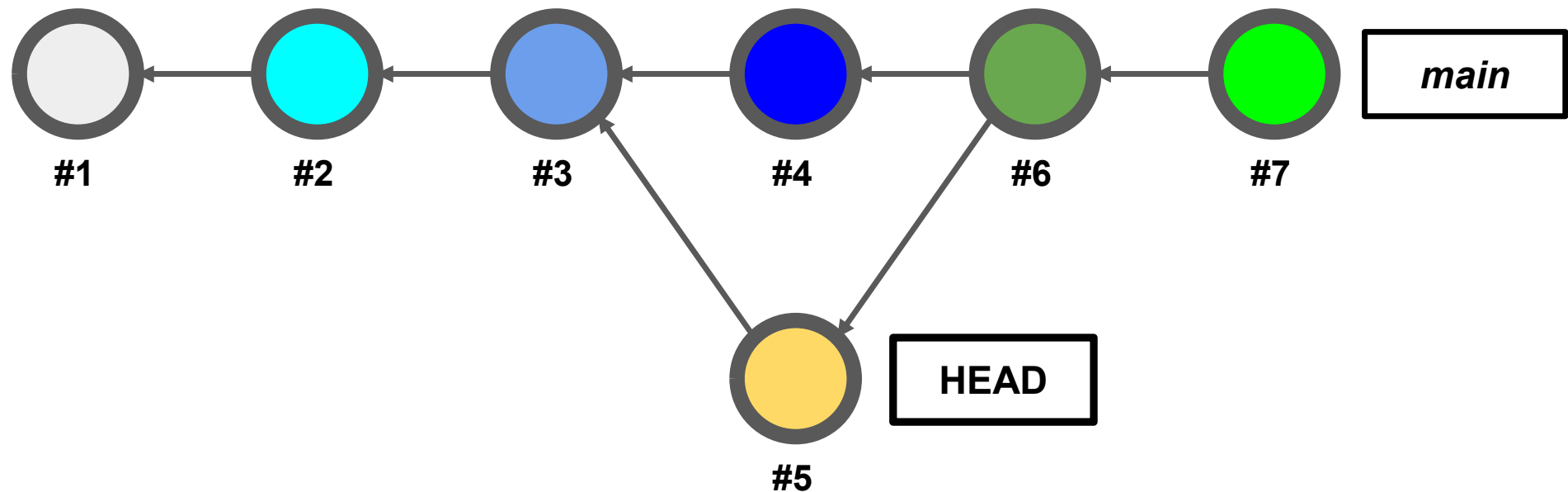
Commit history - HEAD moves as you checkout versions

INSIDE REPO:

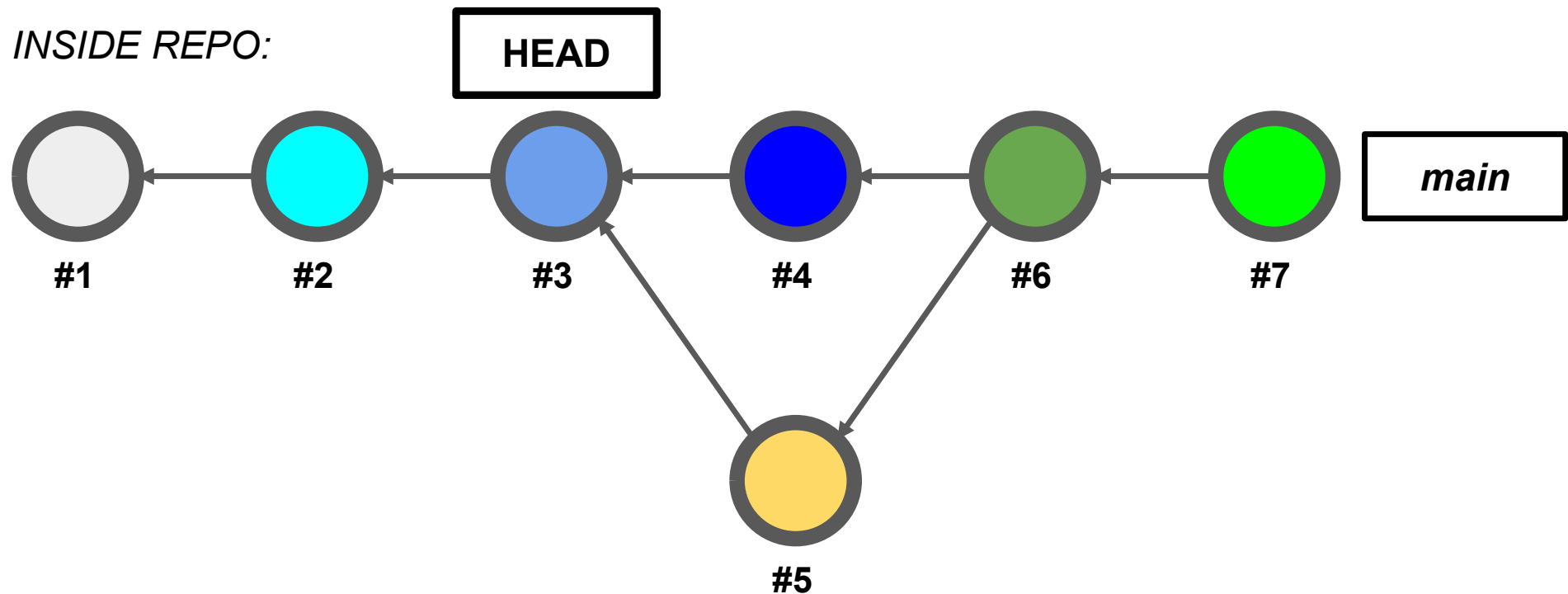


Commit history - HEAD moves as you checkout versions

INSIDE REPO:

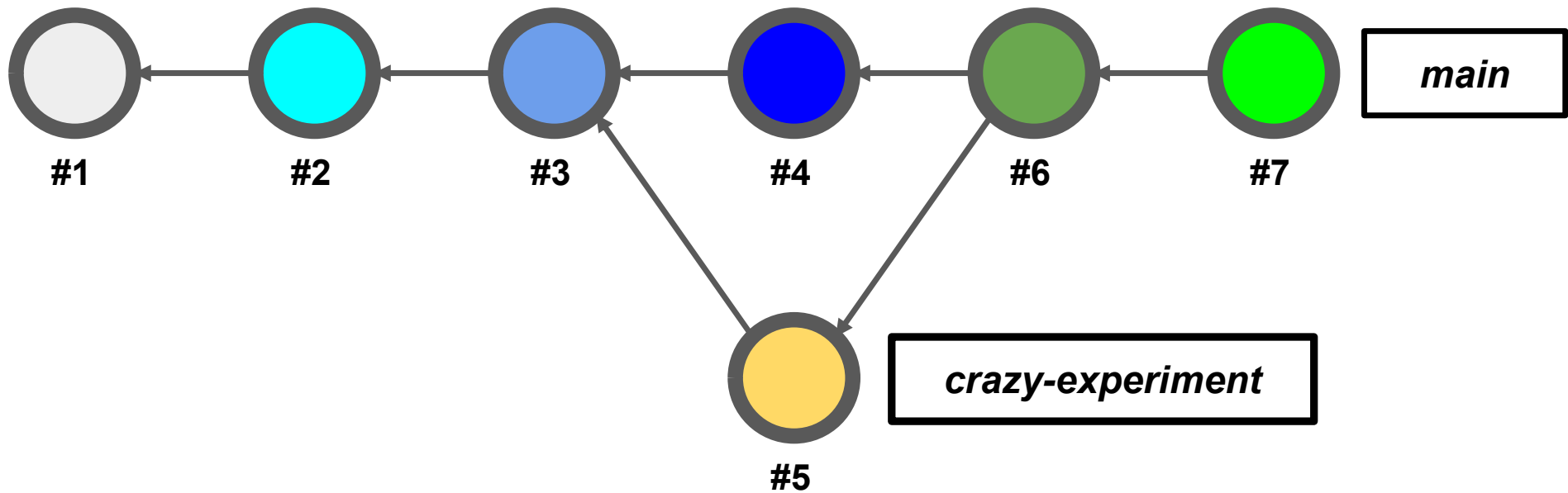


Commit history - HEAD moves as you checkout versions



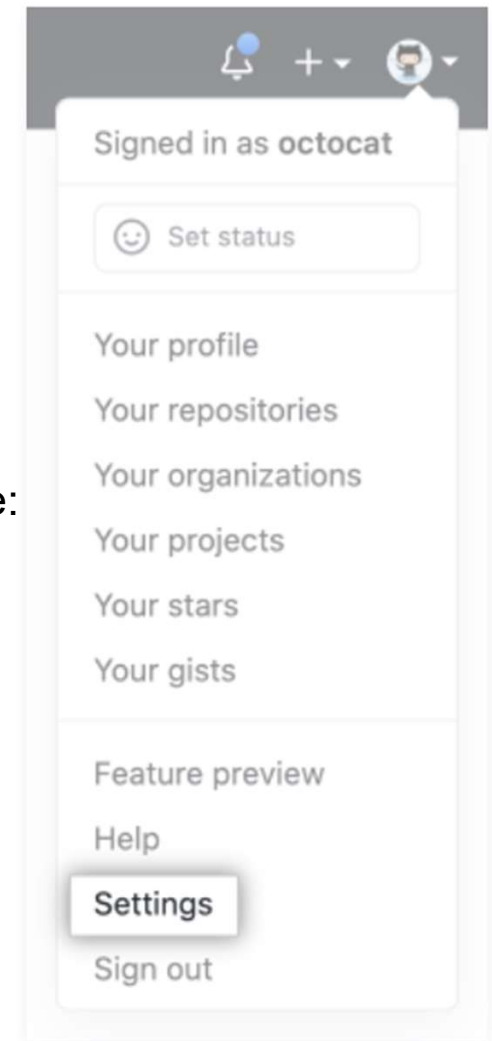
Commit history - you can 'tag' special versions

INSIDE REPO:



GO TO GITHUB.COM AND SIGN-IN:

- Access
- Billing and plans
 - Emails**
 - Password and authentication
 - SSH and GPG keys
 - Organizations
 - Moderation
- Code, planning, and automation
- Repositories
 - Dark themes
- 1) In top-right, select avatar image and choose 'Settings' from dropdown
 - 1) On left side, select 'Emails'
 - 1) Scroll down to make sure the 'Keep my email addresses private' option is checked
 - 1) Copy the email address that looks something like:
223423+username@users.noreply.github.com
... paste it somewhere safe - you might need it soon



☒ Keep my email addresses private

We'll remove your public profile email and use 583231+octocat@users.noreply.github.com when performing web-based Git operations (e.g. edits and merges) and sending email on your behalf. If you want command line Git operations to use your private email you must [set your email in Git](#).

Commits pushed to GitHub using this email will still be associated with your account.

<https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-github-user-account/managing-email-preferences/setting-your-commit-email-address>

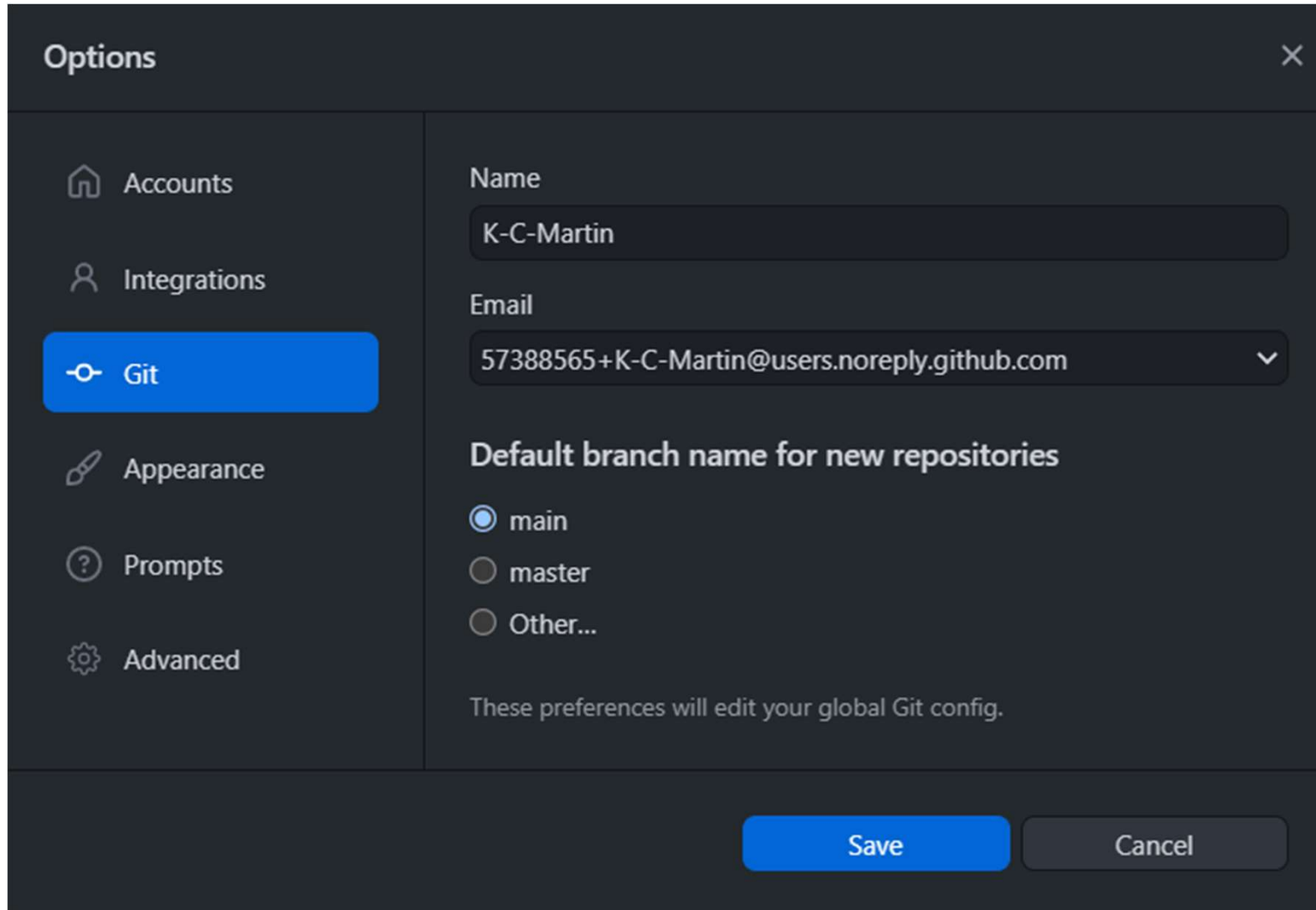
GitHub: [Avatar] > Settings > Emails (*copy email address*)

☒ **Keep my email addresses private**

We'll remove your public profile email and use `57388565+K-C-Martin@users.noreply.github.com` when performing web-based Git operations (e.g. edits and merges) and sending email on your behalf. If you want command line Git operations to use your private email you must [set your email in Git](#).

Commits pushed to GitHub using this email will still be associated with your account.

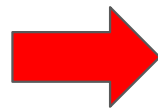
GitHub Desktop: File > Options > Git (*'users.noreply' email*)




The screenshot shows the 'Options' dialog box in GitHub Desktop, with the 'Git' tab selected. The dialog has a dark theme. On the left is a sidebar with icons and labels for 'Accounts', 'Integrations', 'Git' (highlighted in blue), 'Appearance', 'Prompts', and 'Advanced'. The main area on the right contains the following settings:

- Name:** A text field containing 'K-C-Martin'.
- Email:** A dropdown menu showing '57388565+K-C-Martin@users.noreply.github.com' with a downward arrow.
- Default branch name for new repositories:** Three radio button options: 'main' (selected), 'master', and 'Other...'.
- Footer:** A note stating 'These preferences will edit your global Git config.' and two buttons at the bottom right: 'Save' (blue) and 'Cancel' (gray).

CREATE A REPO ON GITHUB





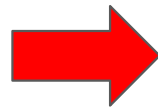
Owner * Repository name *

 K-C-Martin / eucxylo_retreat_2022 ✓

Great repository names are short and memorable. Need inspiration? How about [musical-sniffle?](#)

Description (optional)

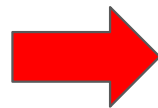
- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.



Initialize this repository with:

Skip this step if you're importing an existing repository.

- ☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)



Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

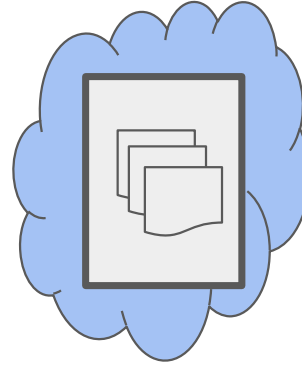
.gitignore template: R ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

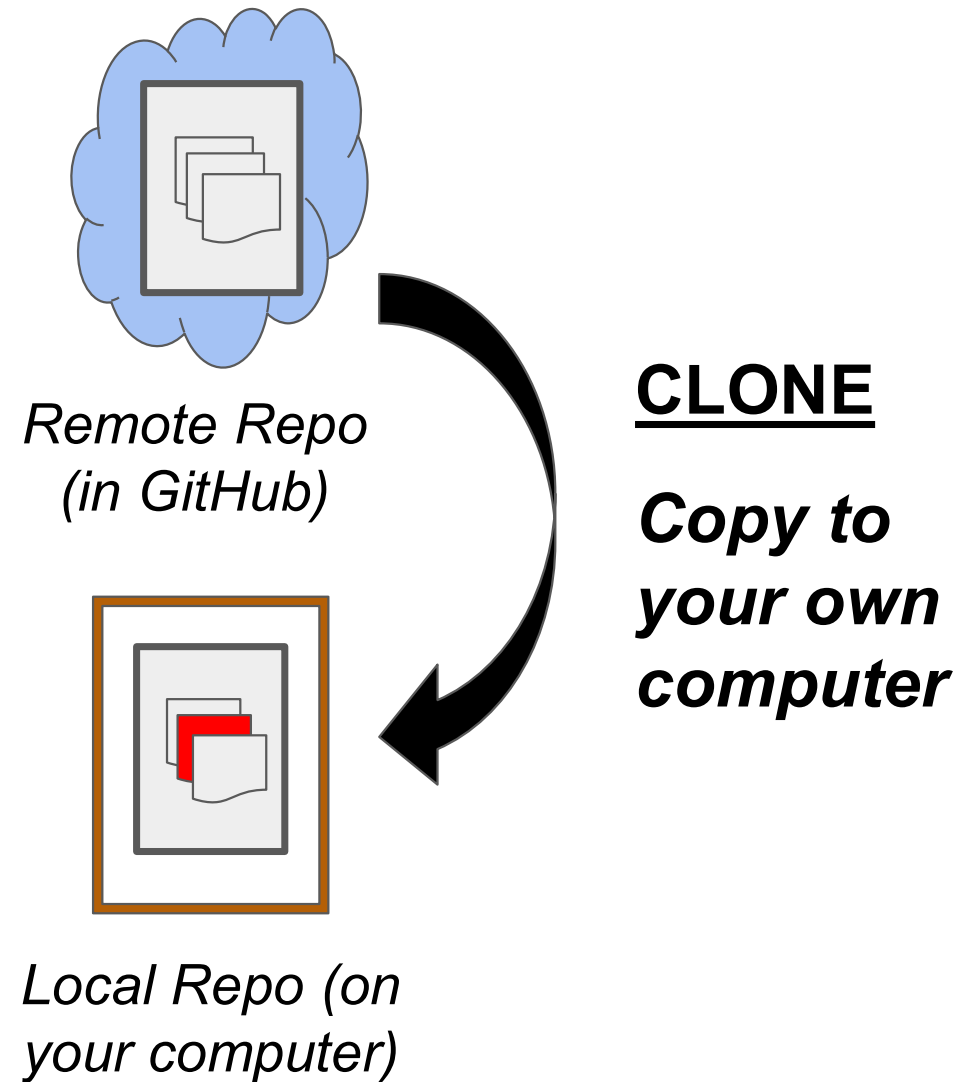
License: None ▾

Creating a Repo on GitHub

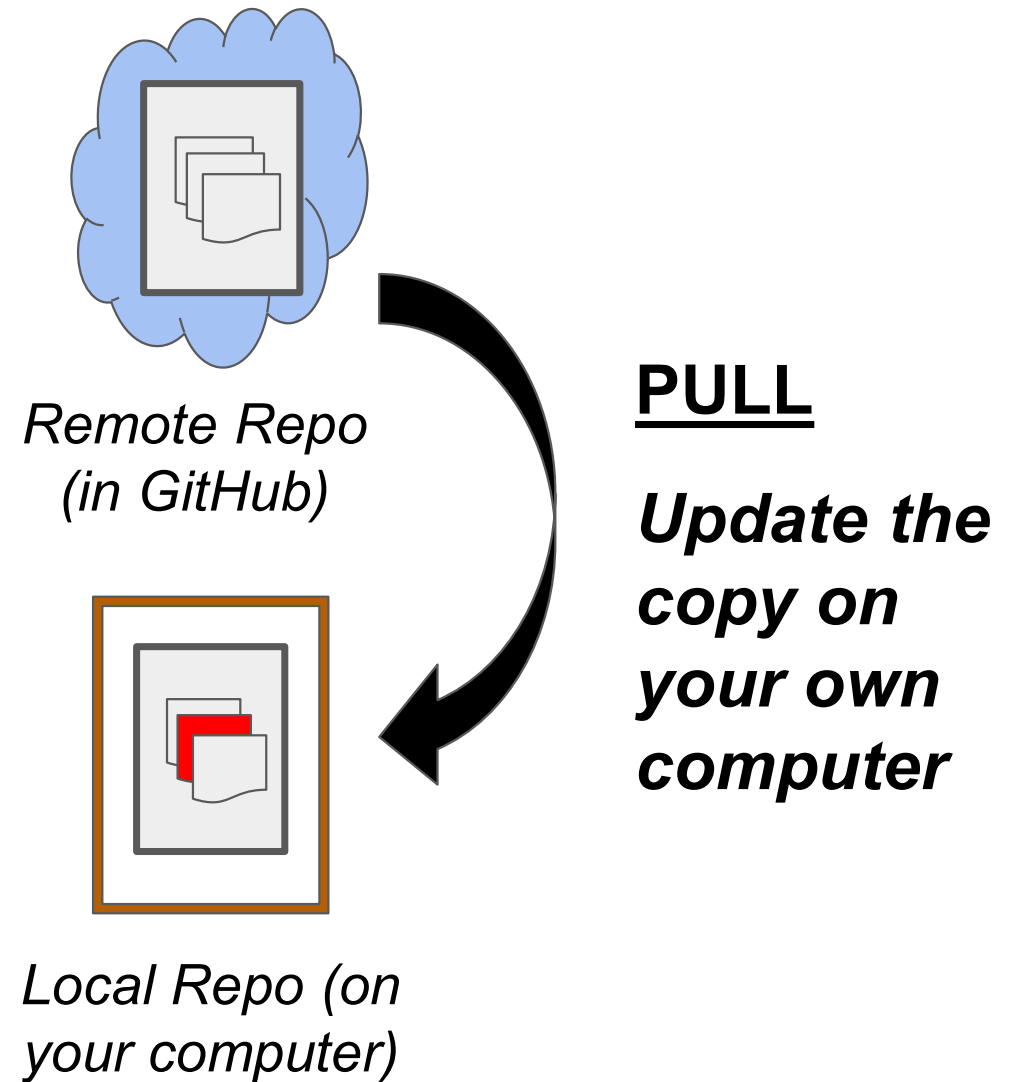


*Remote Repo
(in GitHub)*

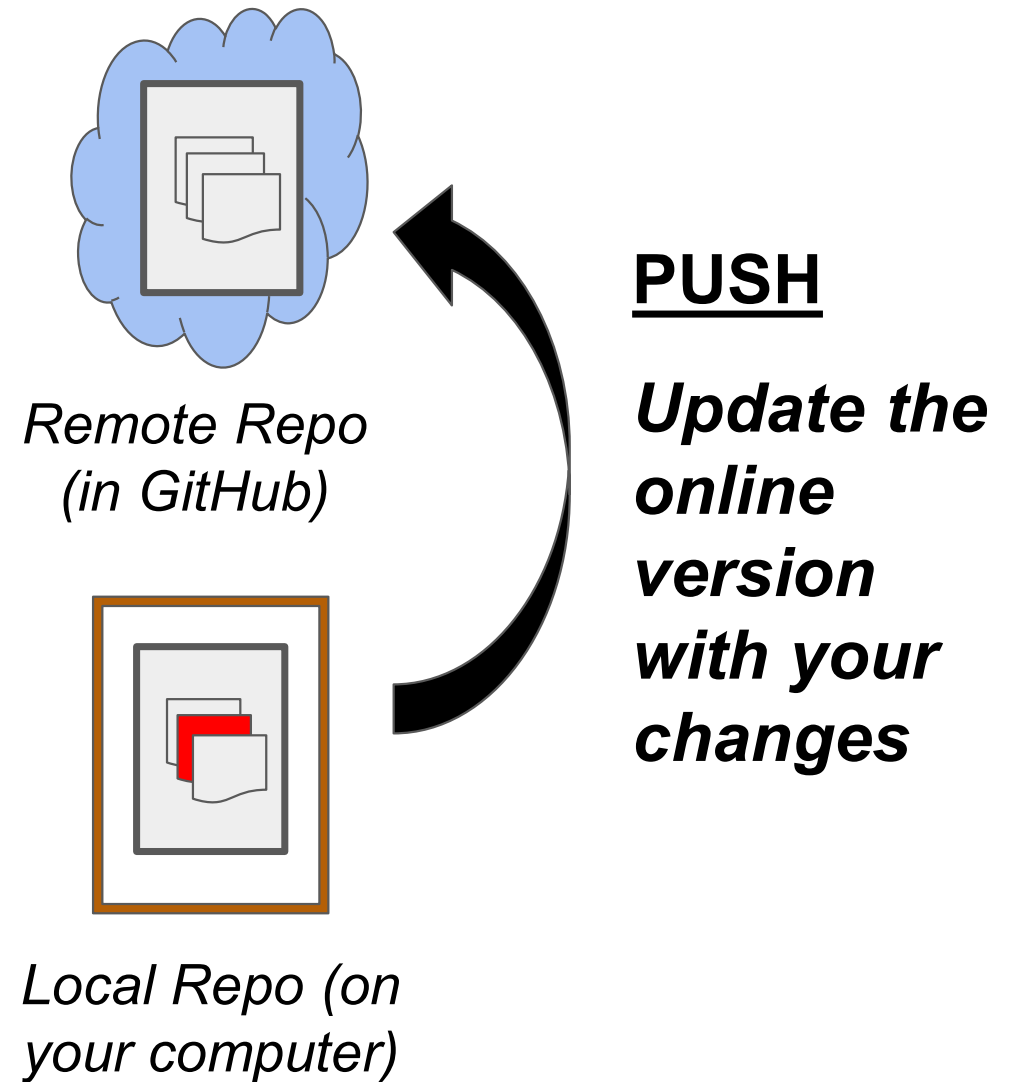
Cloning a repo



Pulling changes to your computer



Pushing changes back to GitHub





Edit the README in your browser... **Commit** a change

Commit changes

Update 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 changes

Cancel

Have a look at your **Commit History**

 main ▾





 1 branch

 0 tags

Go to file

Add file ▾

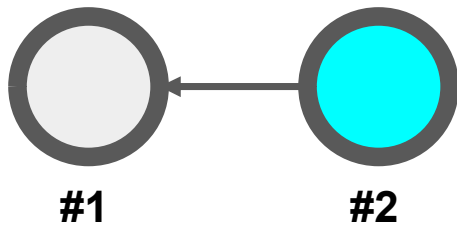
Code ▾

	K-C-Martin Update README.md	253ca3a 4 minutes ago	 2 commits
	.gitignore	Initial commit	1 hour ago
	README.md	Update README.md	4 minutes ago

Open it, then click on the newest commit to see changes

Commit history - timeline of version changes (with IDs)

INSIDE REPO:

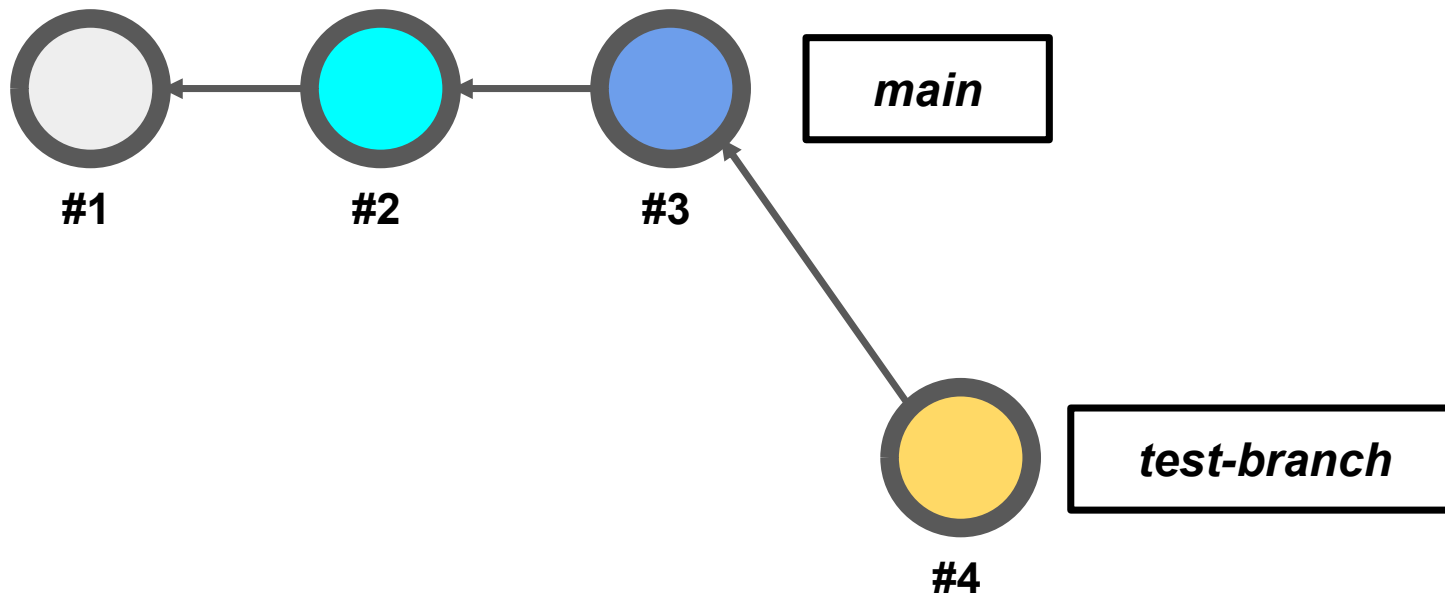


It's a good idea to write short, informative Commit messages.

Try to be 'helpful' to your future self, and your collaborators!

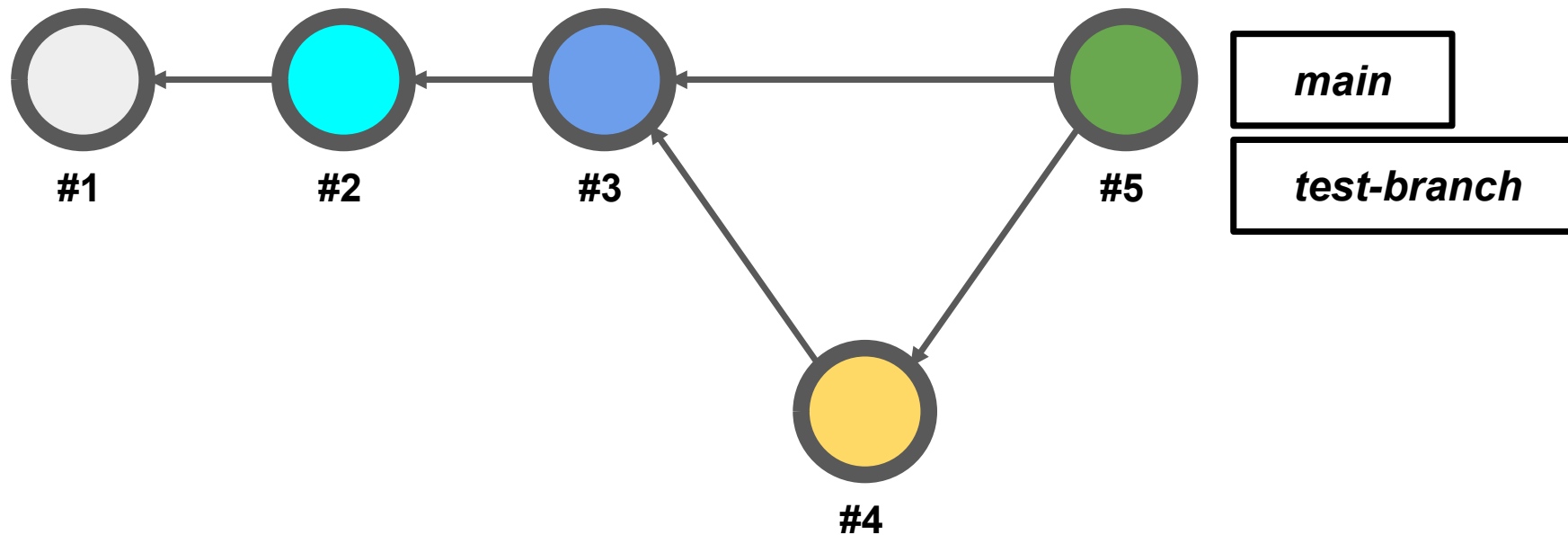
Create branches to test out things...

INSIDE REPO:



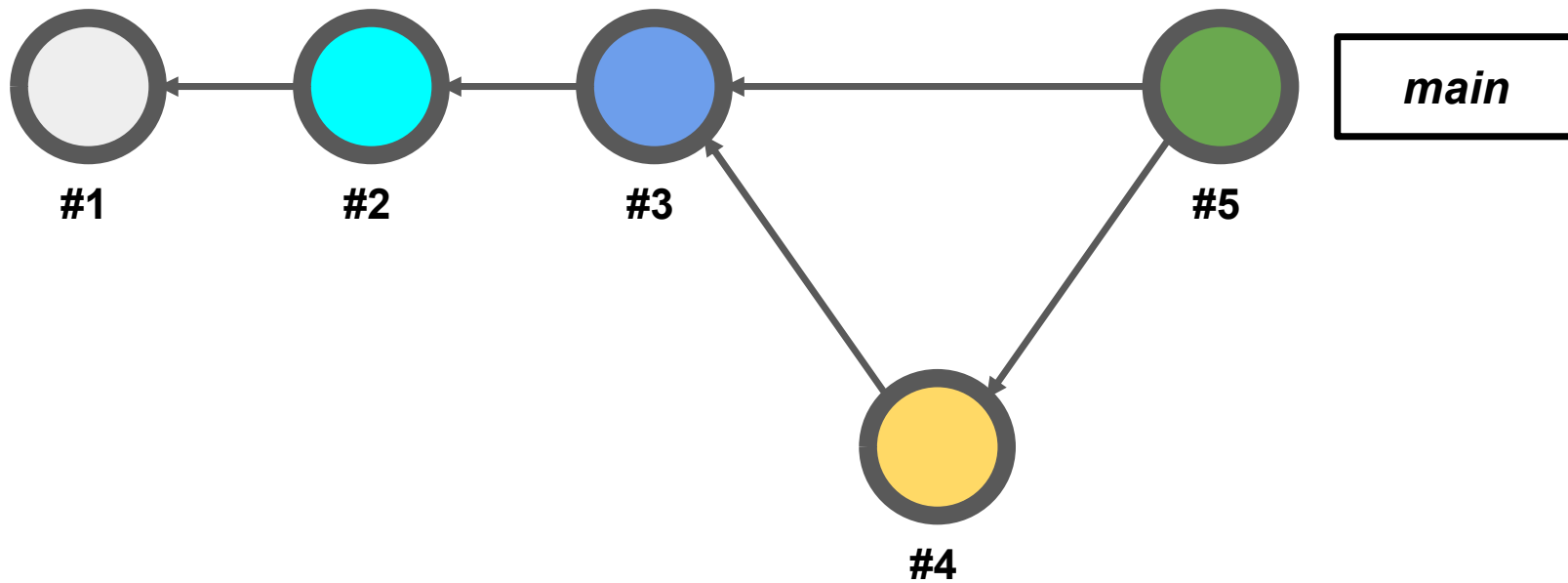
... merge branches into the 'main branch' to add changes

INSIDE REPO:

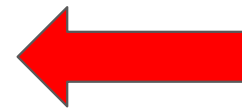
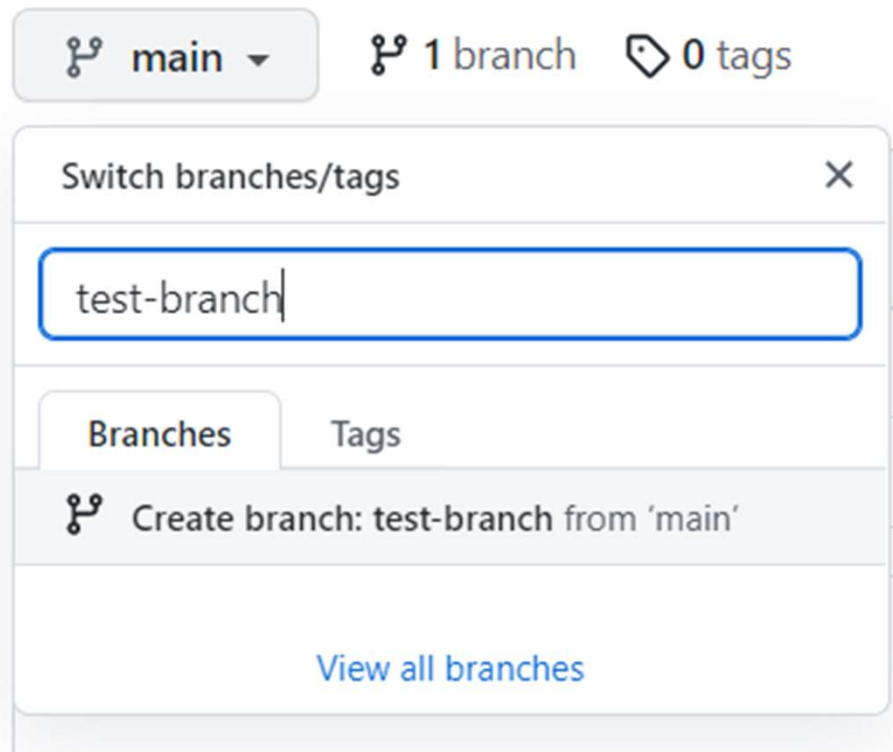


... and delete the reference to the test branch afterwards

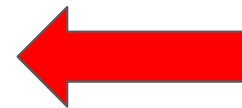
INSIDE REPO:



Create a new branch (call it anything you want)





Give the new branch a name




Then create it

EXERCISE: Edit a file in the new branch

- In your new branch, edit the README again 
- Give it a good commit message!
- Look at the commit history  3 commits
- Switch back to the main branch - compare the two




Commit changes

Update README.md

 main ▼


 2 branches




Open a Pull Request (PR), merge the branches

 main ▾  2 branches  Click to view all branches





Search branches...

Overview Yours Active Stale All branches New branch






Default branch 

 main  Updated 12 minutes ago by K-C-Martin Default 

Your branches


 test-branch  Updated 41 seconds ago by K-C-Martin 0 | 1  New pull request  **PR**


Active branches

 test-branch  Updated 41 seconds ago by K-C-Martin 0 | 1  New pull request  

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

 base: main ◯ ← compare: test-branch ◯ ✓ **Able to merge.** These branches can be automatically merged.



Another change


Write

Preview

H B I ≡ <> 🔗 ≡ ≡ ☑ @ ↗ ↶

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.



Create pull request

Have a look at the 'Pull requests' tab in your repo

[Code](#) [Issues](#) [Pull requests 1](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [...](#)

Filters [▼](#)

Labels 9

Milestones 0

New pull request

[1 Open](#) [2 Closed](#)

<input type="checkbox"/>	Author ▼	Label ▼	Projects ▼	Milestones ▼	Reviews ▼	Assignee ▼	Sort ▼
<input type="checkbox"/>	🔗 Another change						
	#3 opened 36 seconds ago by K-C-Martin						

Another change #1



K-C-Martin wants to merge 1 commit into `main` from `test-branch`



Conversation 0



Commits 1



Checks 0



Files changed 1



K-C-Martin commented now



No description provided.



Another change

Verified

bbc872b

Add more commits by pushing to the `test-branch` branch on K-C-Martin/2022-06_DCW.



Continuous integration has not been set up

[GitHub Actions](#) and [several other apps](#) can be used to automatically catch bugs and enforce style.



This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request



You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

OR ADD A
COMMENT
BELOW TO ASK
FOR MORE
CHANGES...

Another change #1



K-C-Martin wants to merge 1 commit into `main` from `test-branch`



Conversation 0



Commits 1



Checks 0



Files changed 1



K-C-Martin commented 1 minute ago



No description provided.



Another change

Verified

bbc872b

Add more commits by pushing to the `test-branch` branch on K-C-Martin/2022-06_DCW.



Merge pull request #1 from K-C-Martin/test-branch

Another change

Confirm merge

Cancel




Another change #1

 **Merged** K-C-Martin merged 1 commit into `main` from `test-branch`  now

 Conversation 0

 Commits 1

 Checks 0

 Files changed 1



K-C-Martin commented 3 minutes ago



No description provided.

  Another change

Verified

bbc872b

  K-C-Martin merged commit `04814b9` into `main` now

Revert



Pull request successfully merged and closed

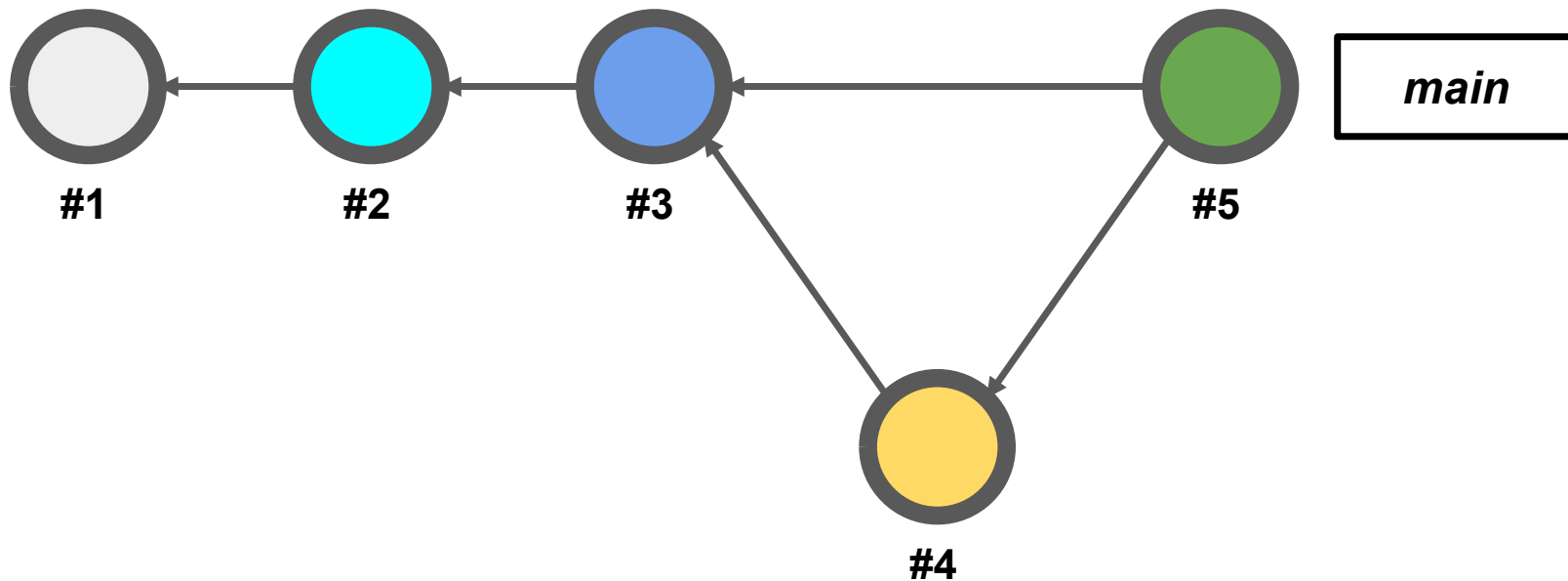
You're all set—the `test-branch` branch can be safely deleted.

Delete branch

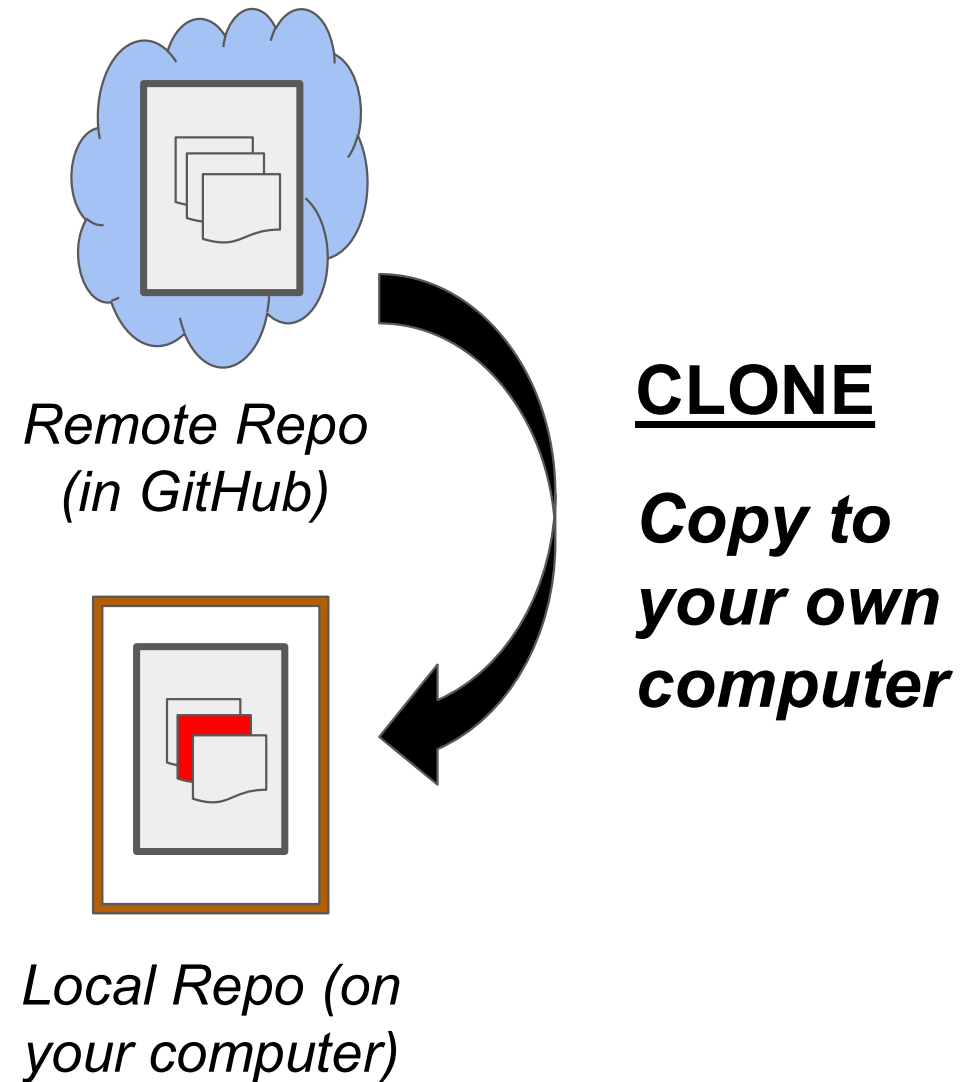
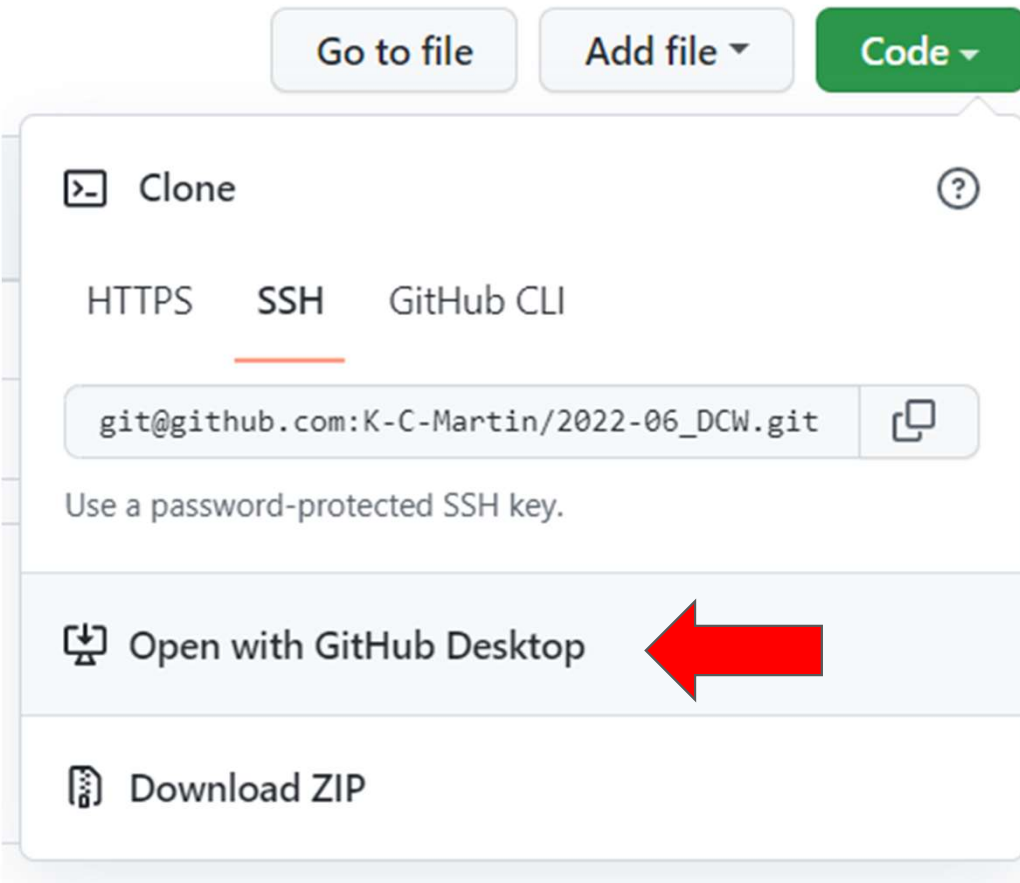


Have a look at your commit history - see the 'merge'

INSIDE REPO:

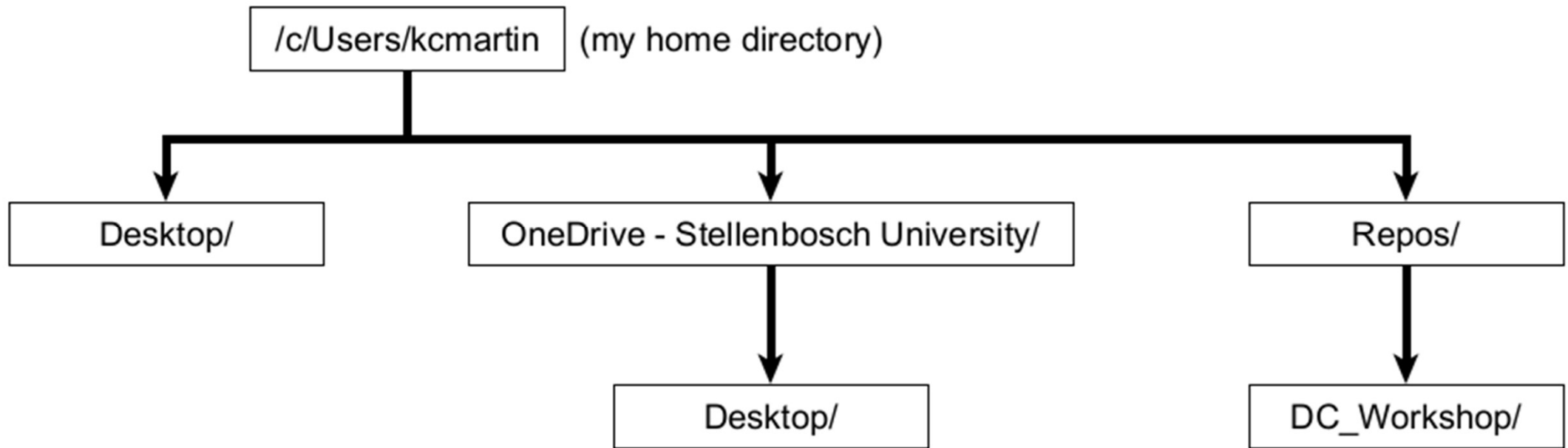


Cloning a repo

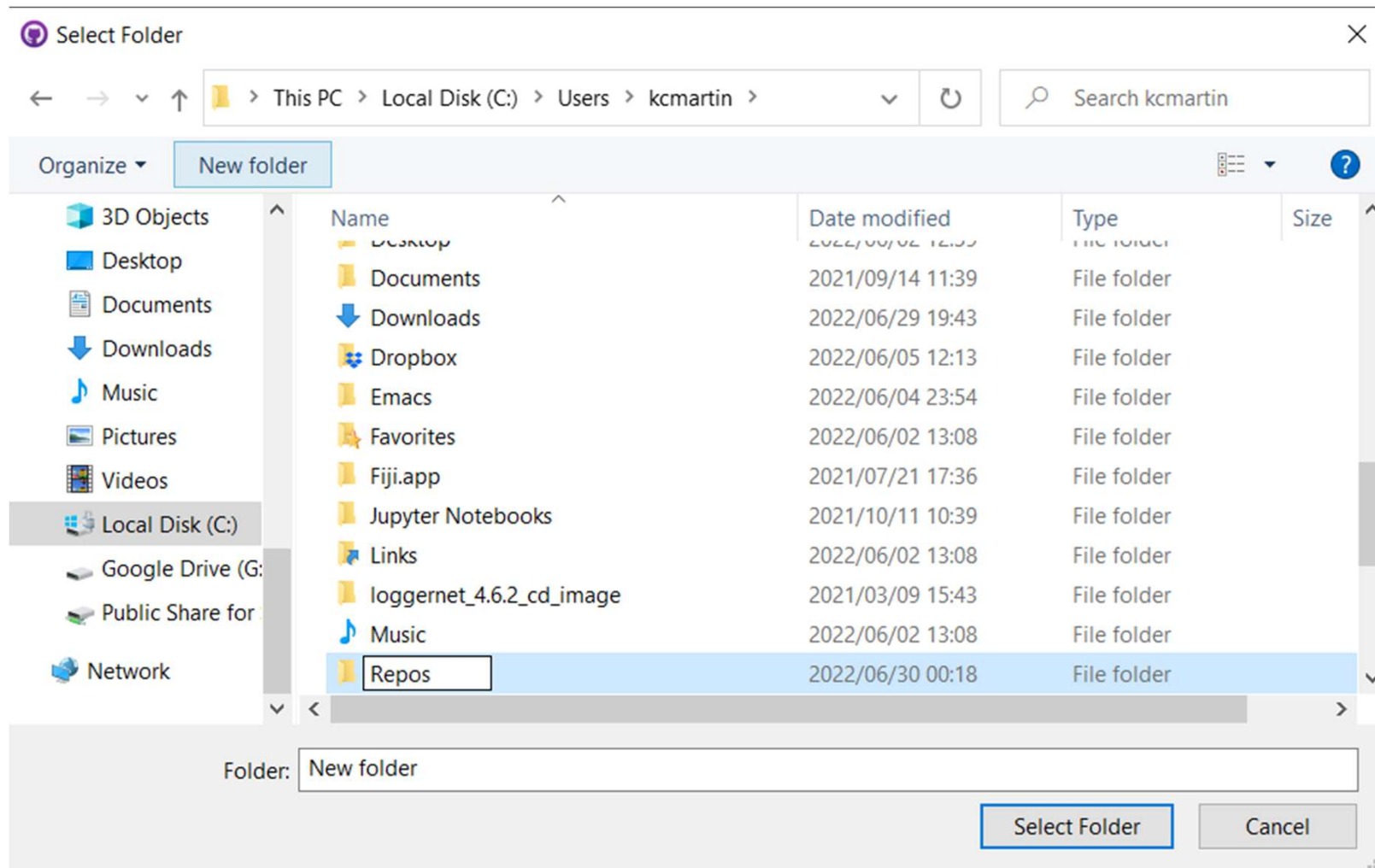


Cloning a repo

Do NOT put your Repos folder in OneDrive!!



Create a 'Repos' folder in your User Directory



Cloning a repo

Clone a repository

GitHub.com GitHub Enterprise **URL**

Repository URL or GitHub username and repository
(hubot/cool-repo)

https://github.com/K-C-Martin/eucxylo_retreat_2022

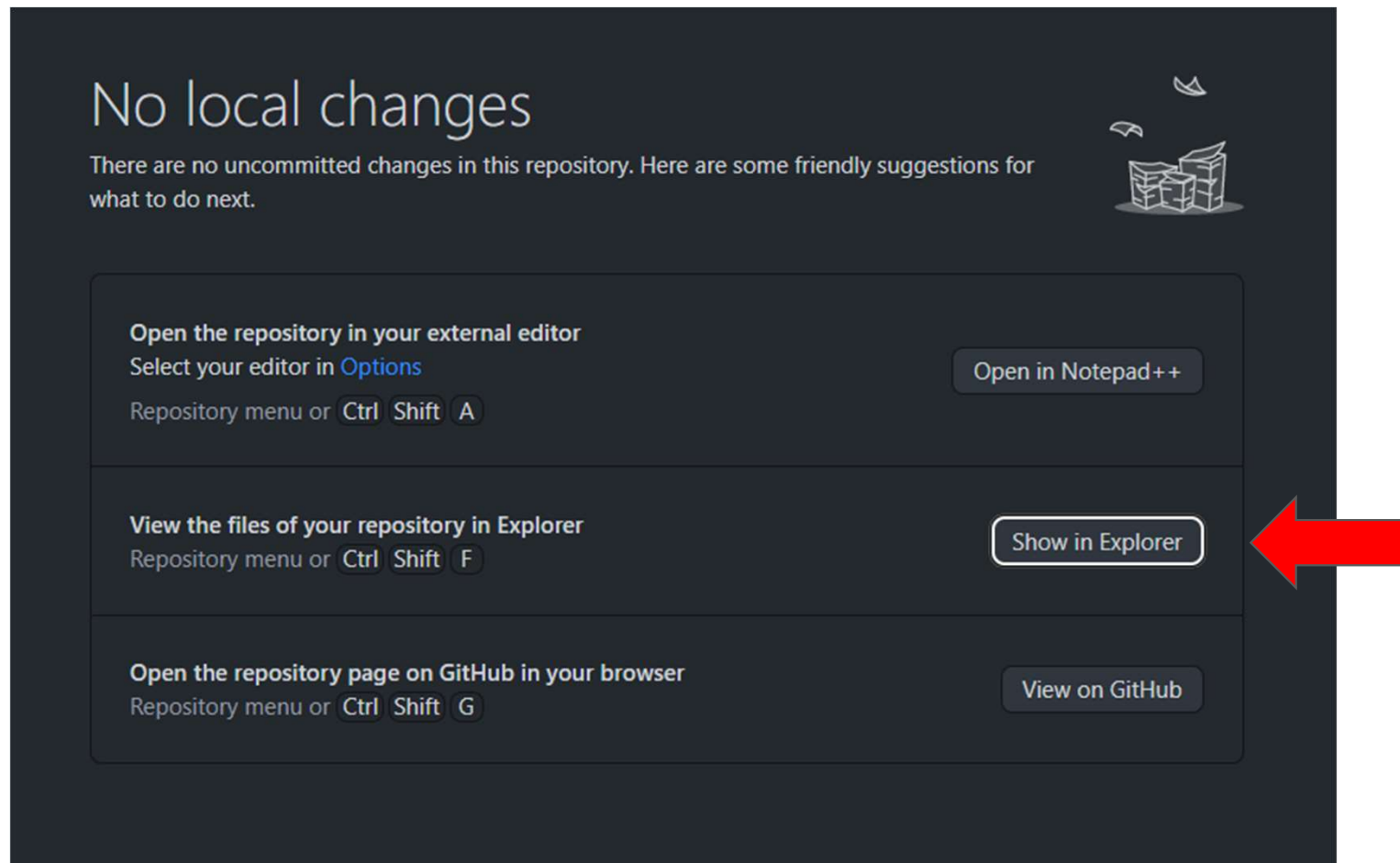
Local path

C:\Users\kcmartin\Repos\eucxylo_retreat_2022 Choose...

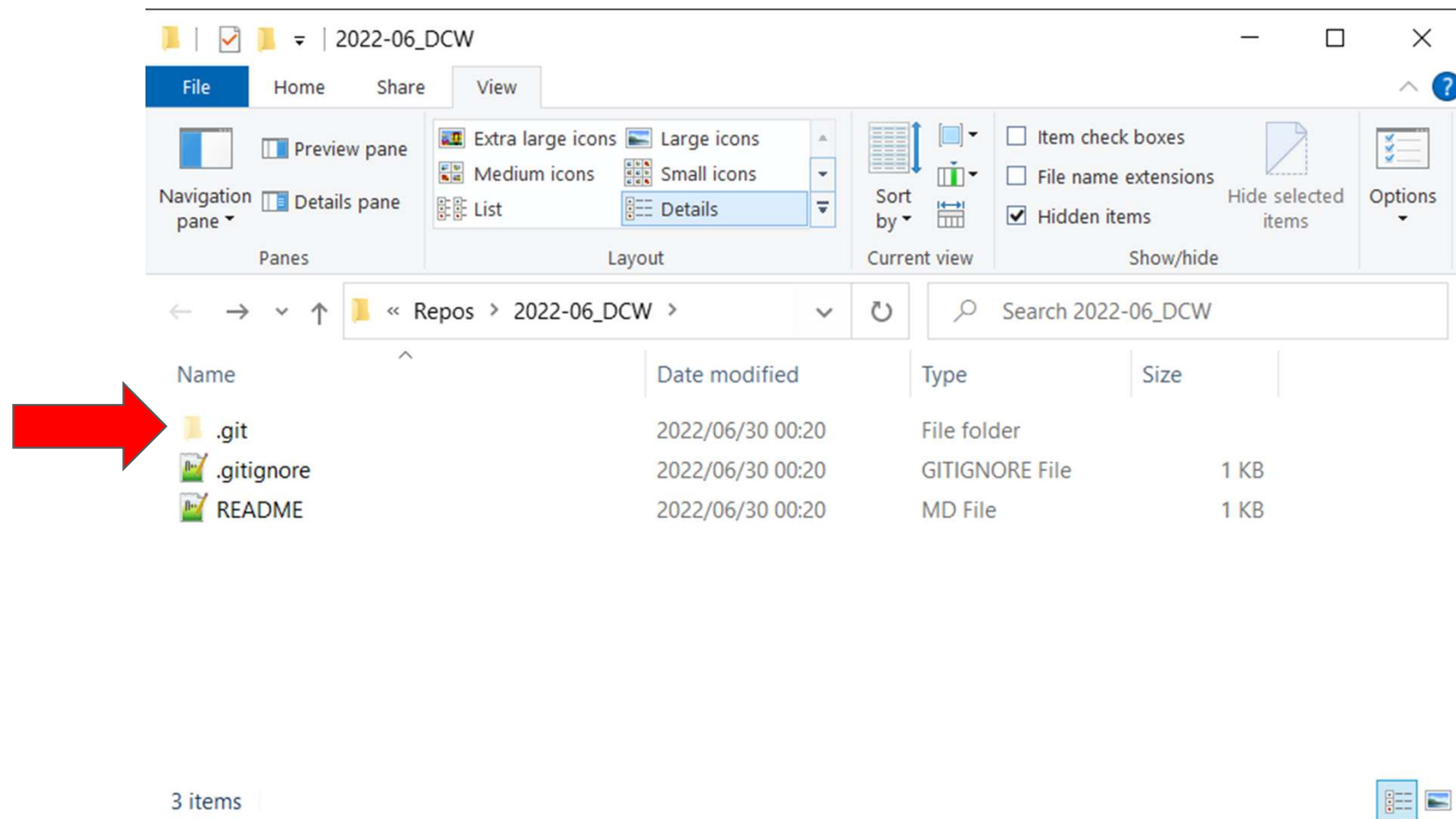
Clone Cancel

**Do NOT put your
Repos folder in
OneDrive!!**

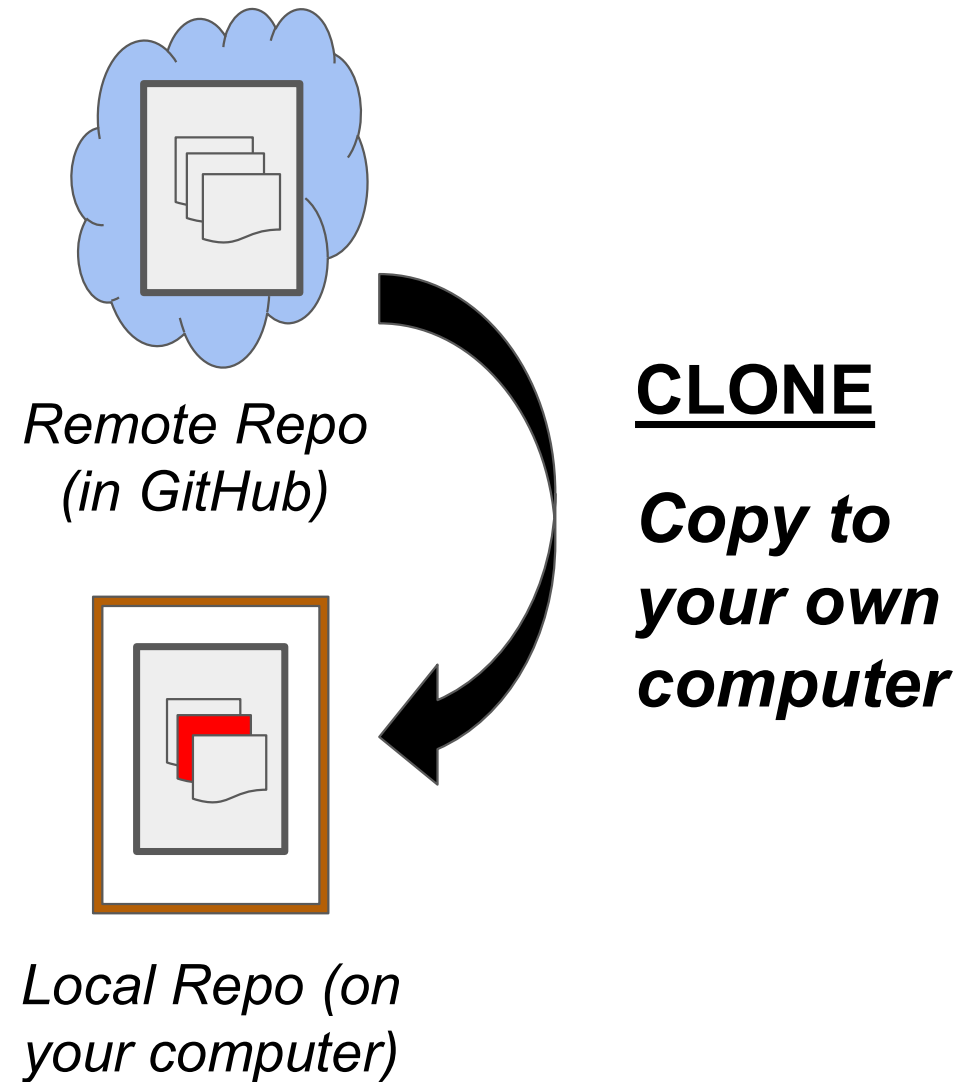
Open the folder containing your Repo



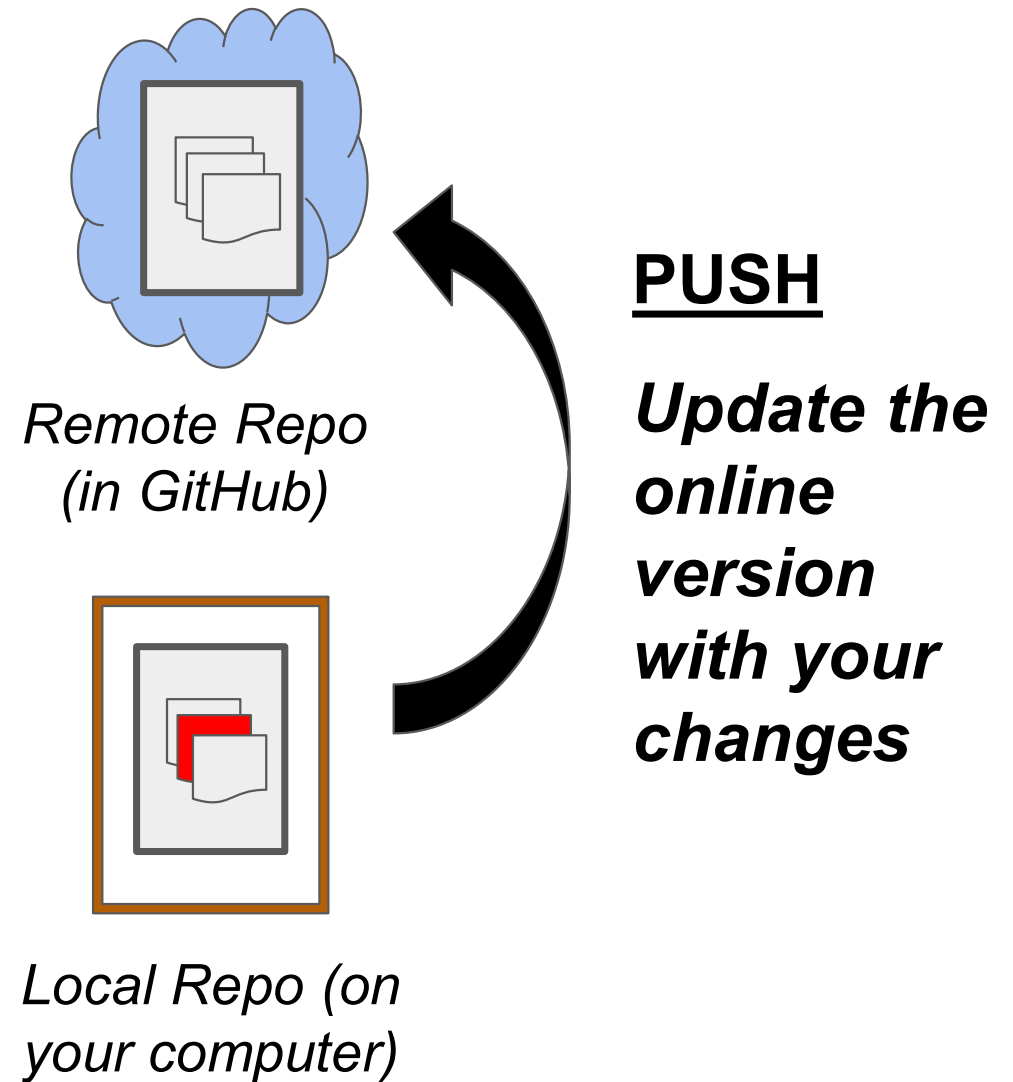
The hidden '.git' folder is your Git Database (all commits)



Cloning a repo

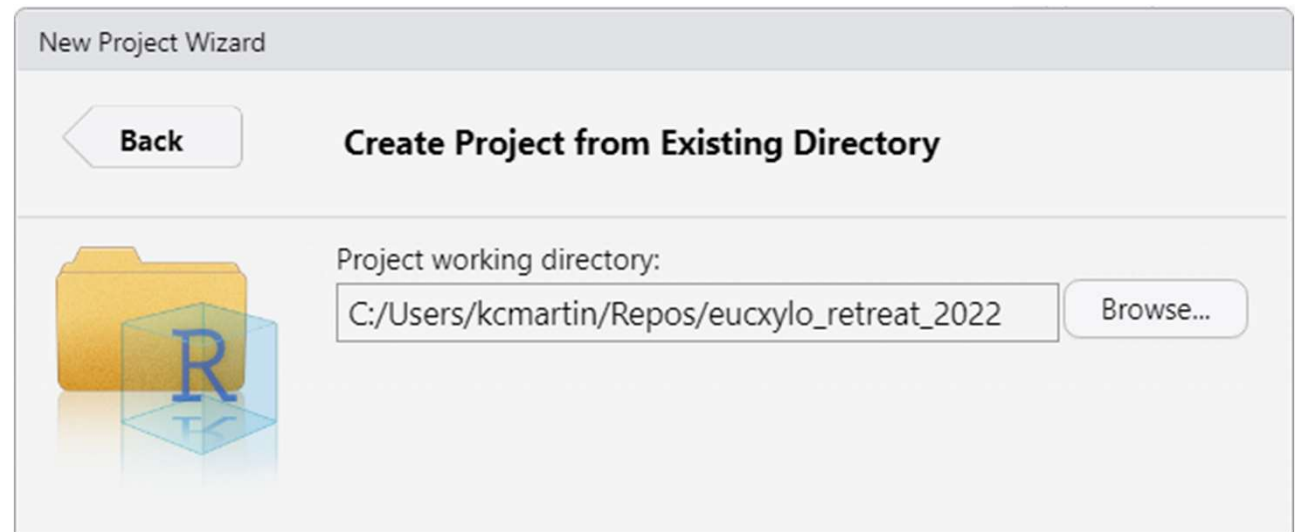


Pushing changes back to GitHub



EXERCISE: Create an RStudio Project in the repo

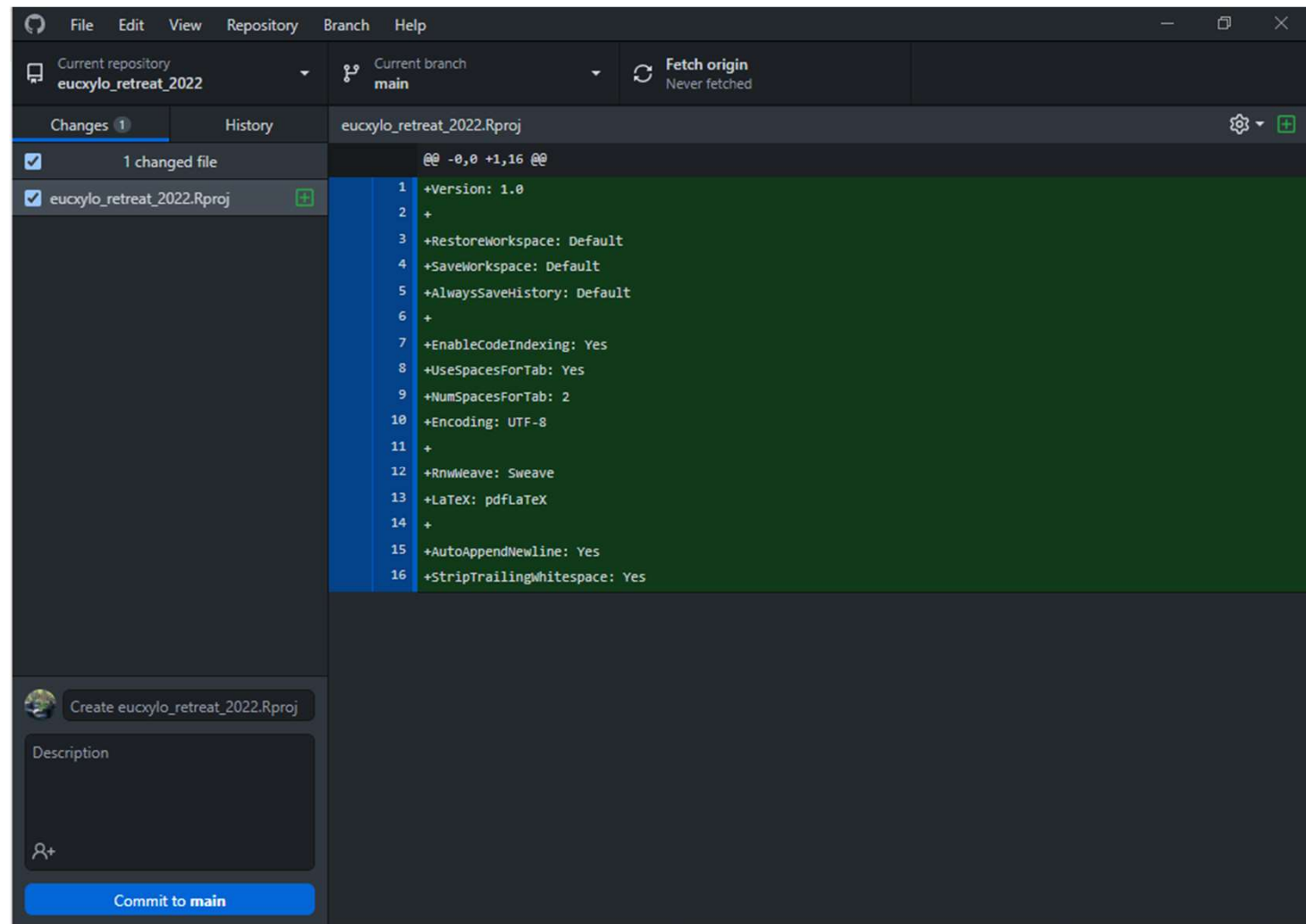
- Open RStudio
- File > New Project
- Create a New Project in an **Existing Directory**
- Pick your new repo folder (look inside the Repos folder in your user directory)



Have a look at GitHub Desktop...

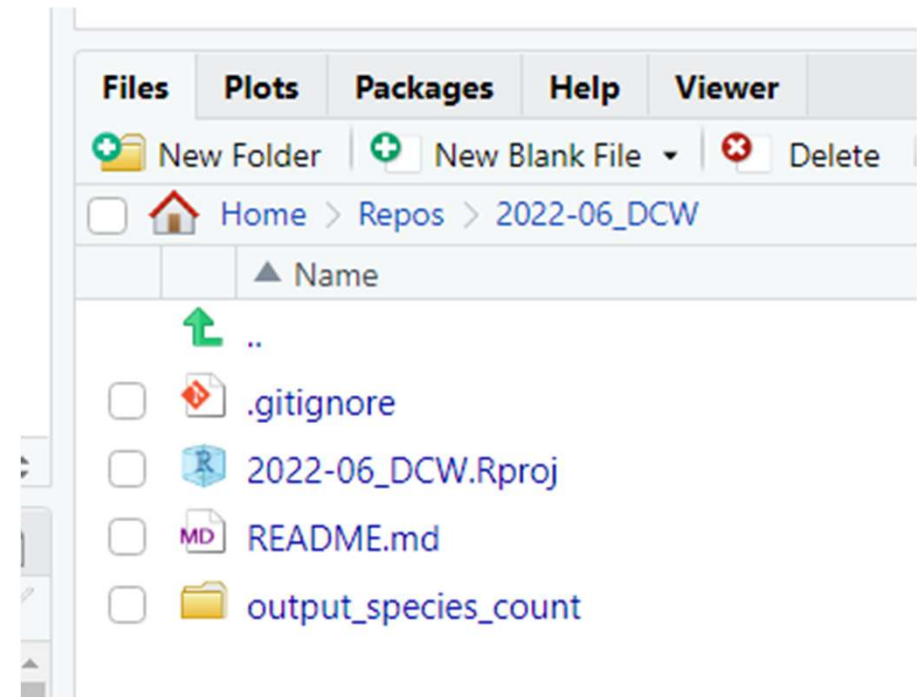
Newly-created 'R Project' file

Note: this hasn't been added to the git 'database' yet... but git (GitHub) is 'aware' of it.



EXERCISE: Create a folder and an R Script file in the repo

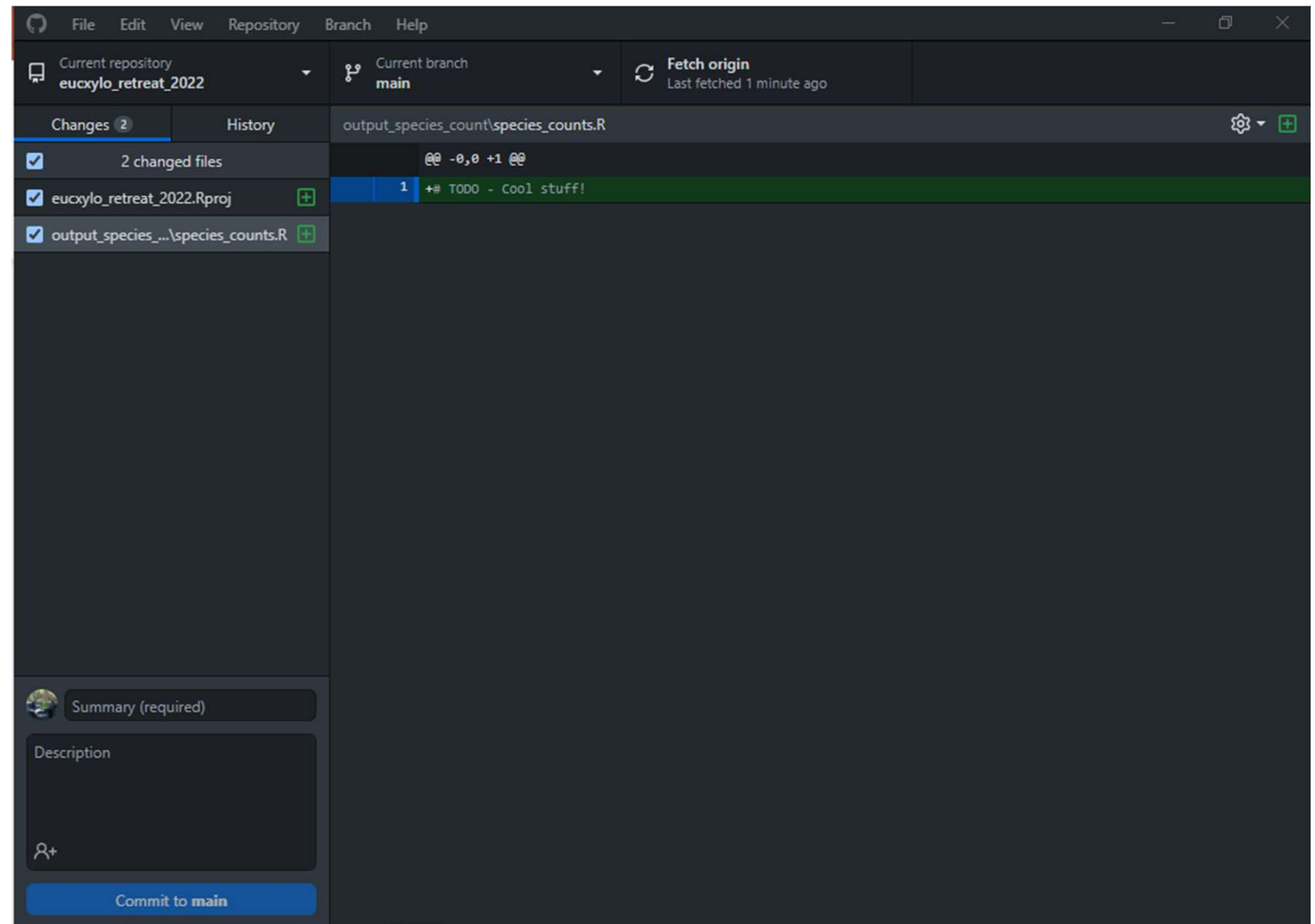
- Create a New Folder in RStudio - call it: **output_species_count**
- File > New File > R Script
- Add some text : **# TODO - Cool stuff!**
- File > Save As ... save the script
in the new folder (*output_species_count*)
as **species_counts**



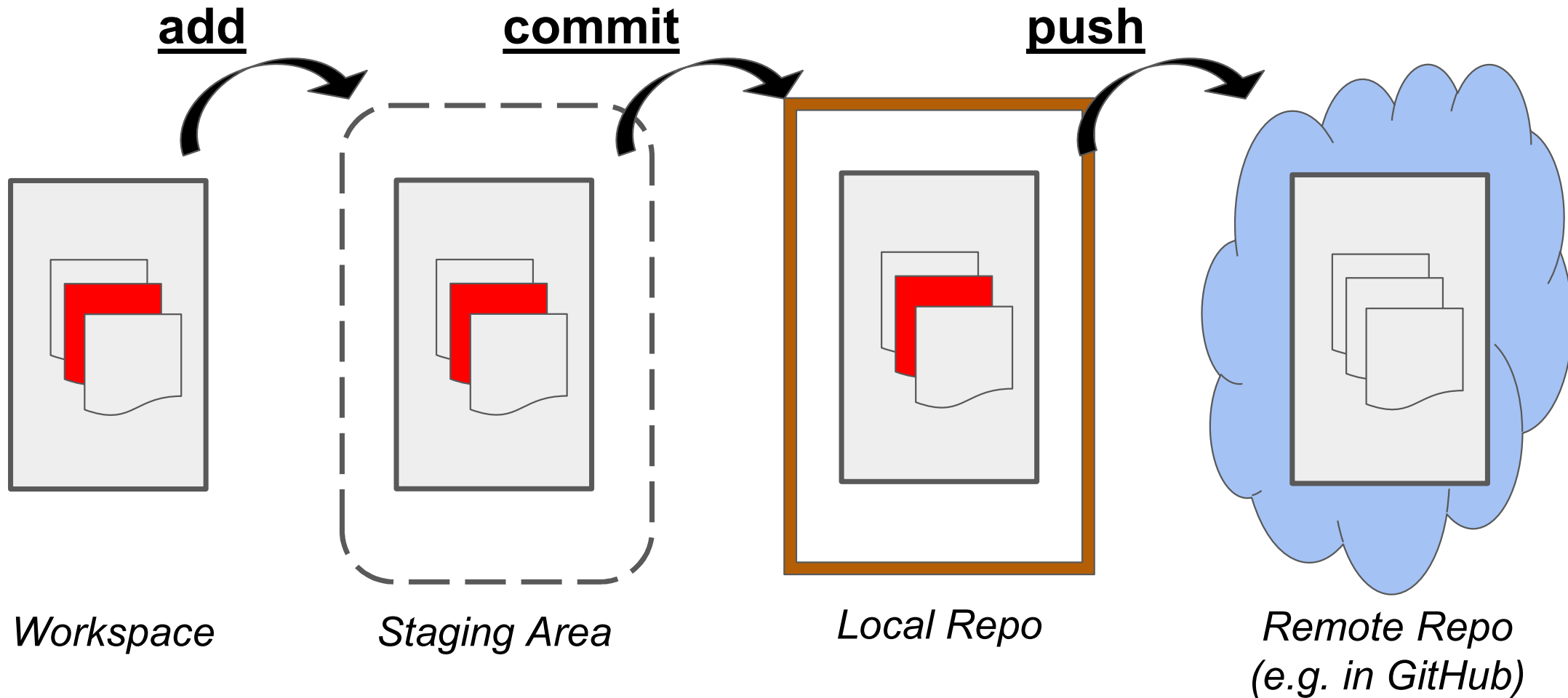
Have a look at GitHub Desktop...

Newly-created 'R' file along with previously-created 'R Project' file

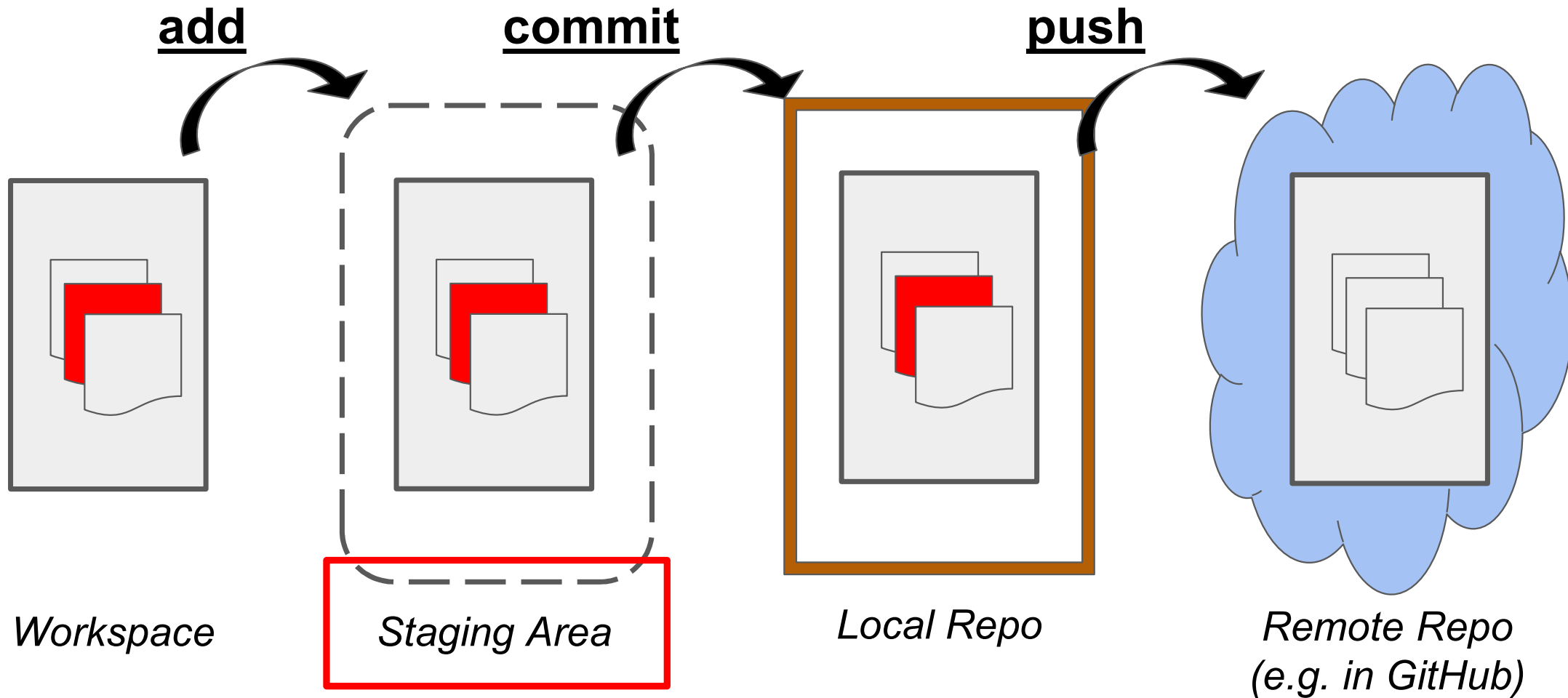
Note: these haven't been added to the git 'database' yet... but git (GitHub) is 'aware' of them.



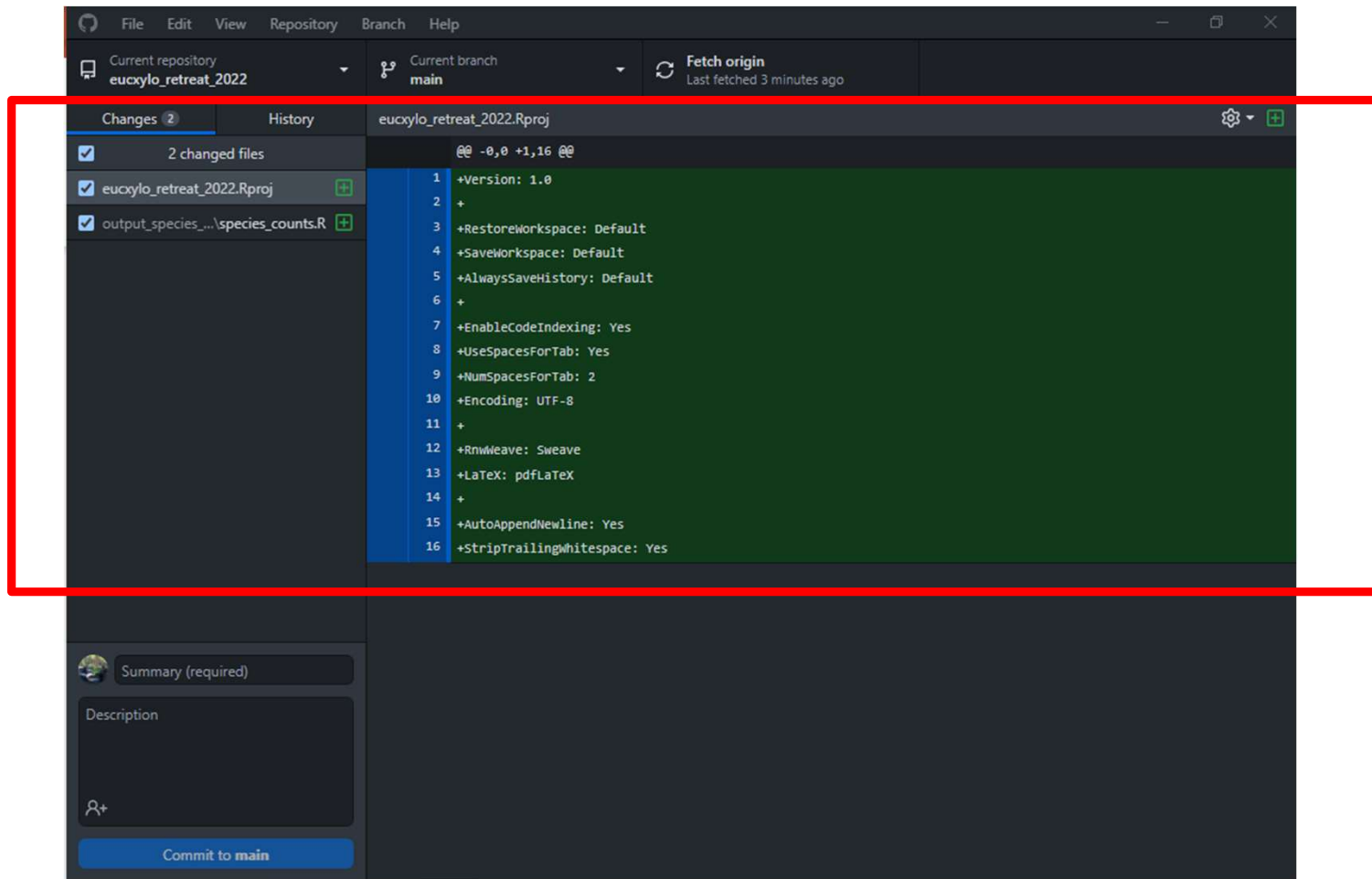
Basic idea - staging and committing



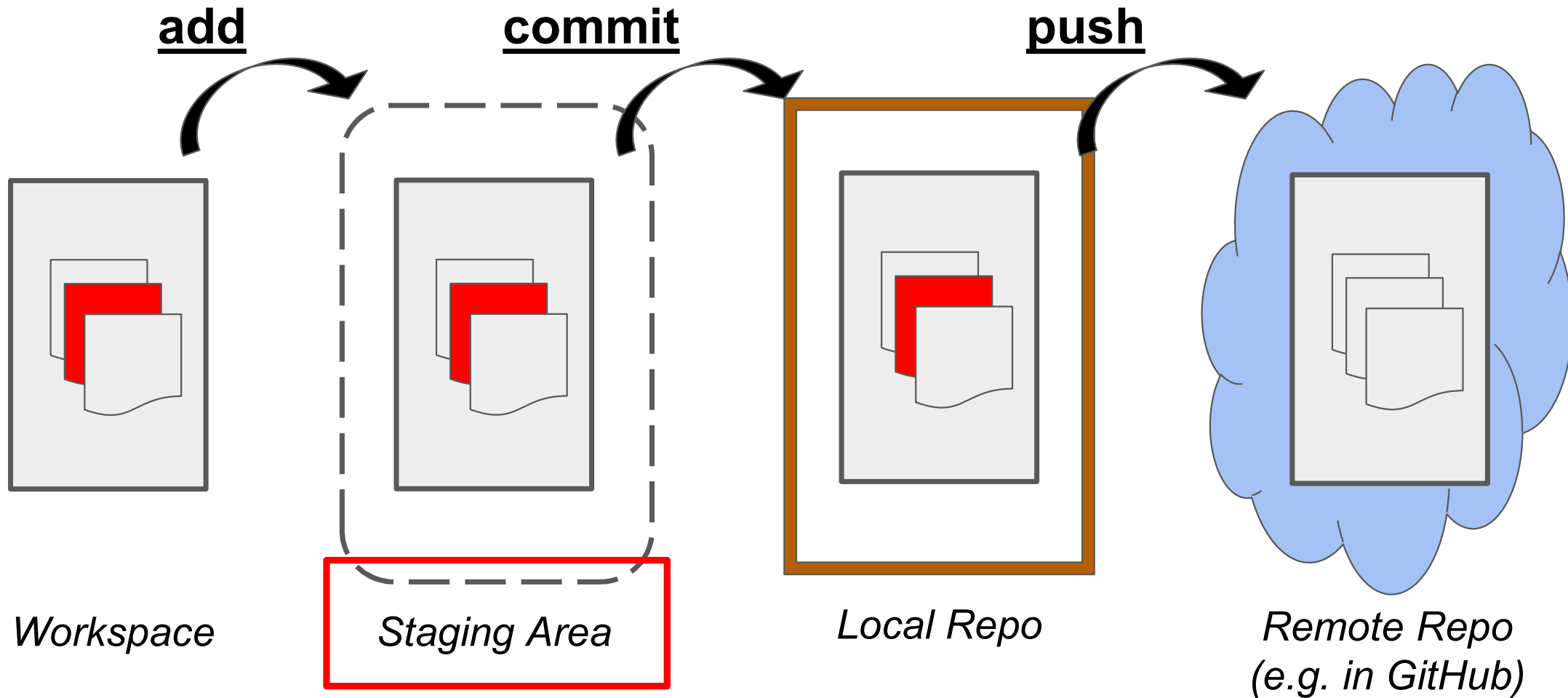
Basic idea - staging and committing



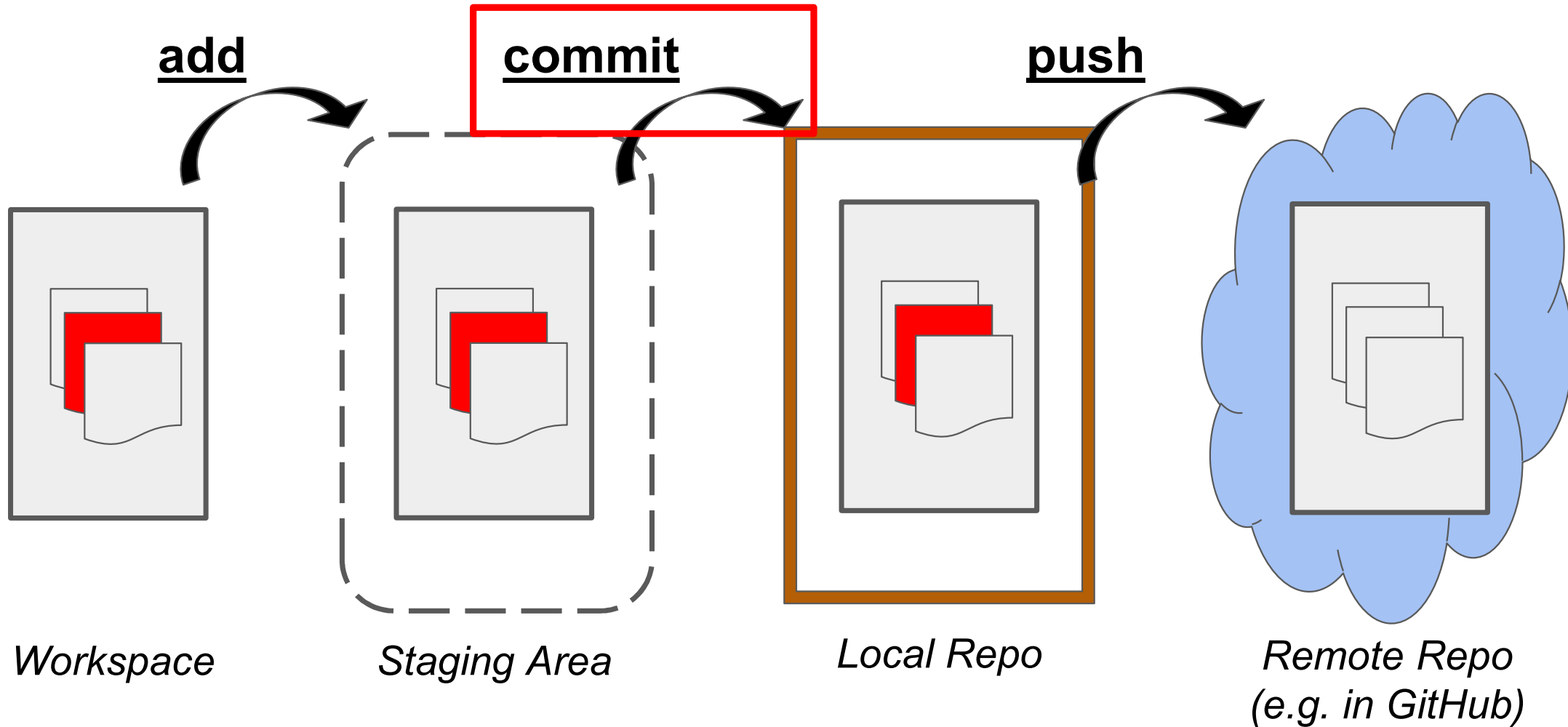
Files that are selected are 'Staged' to be Committed



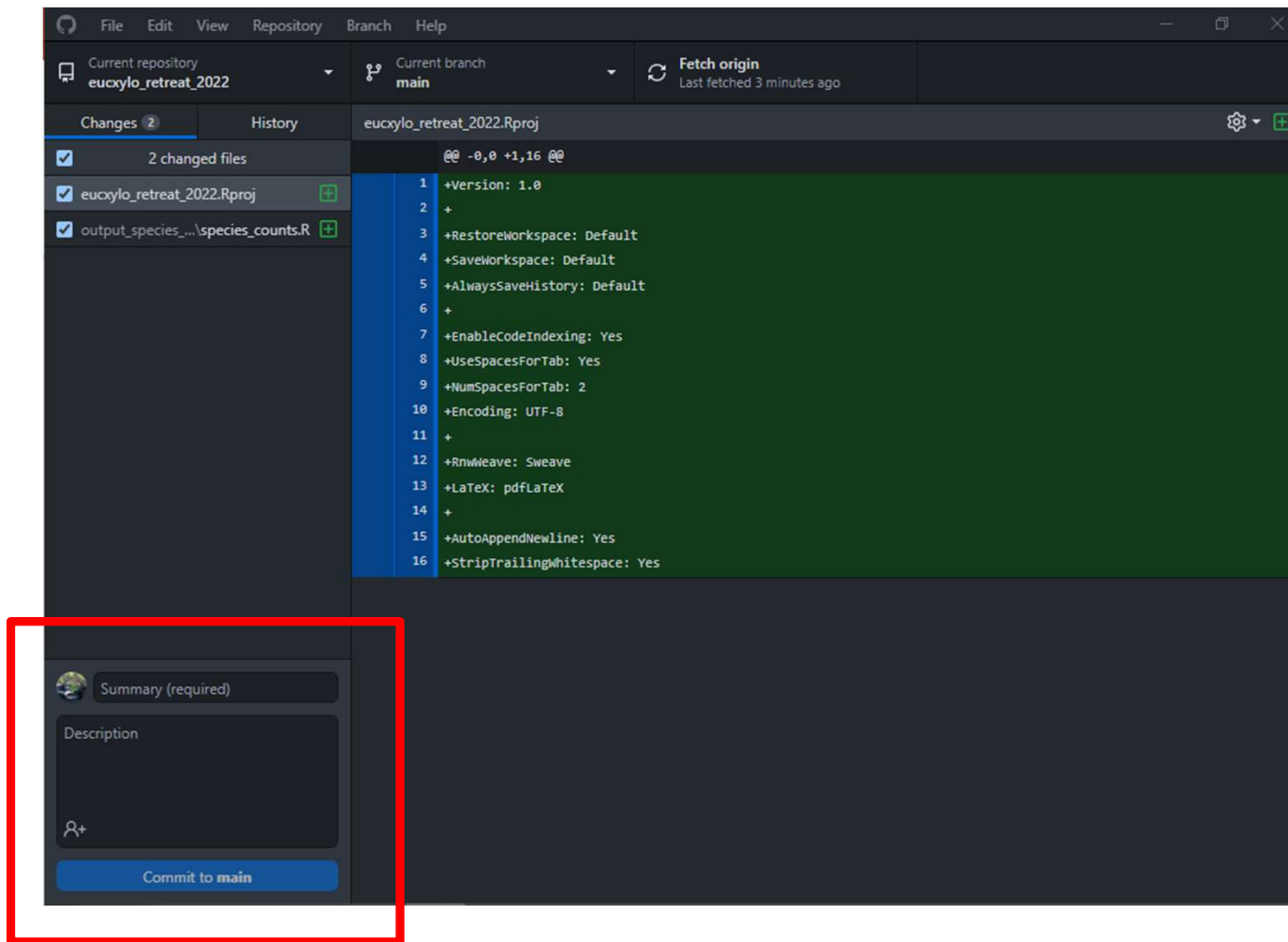
Basic idea - staging and committing



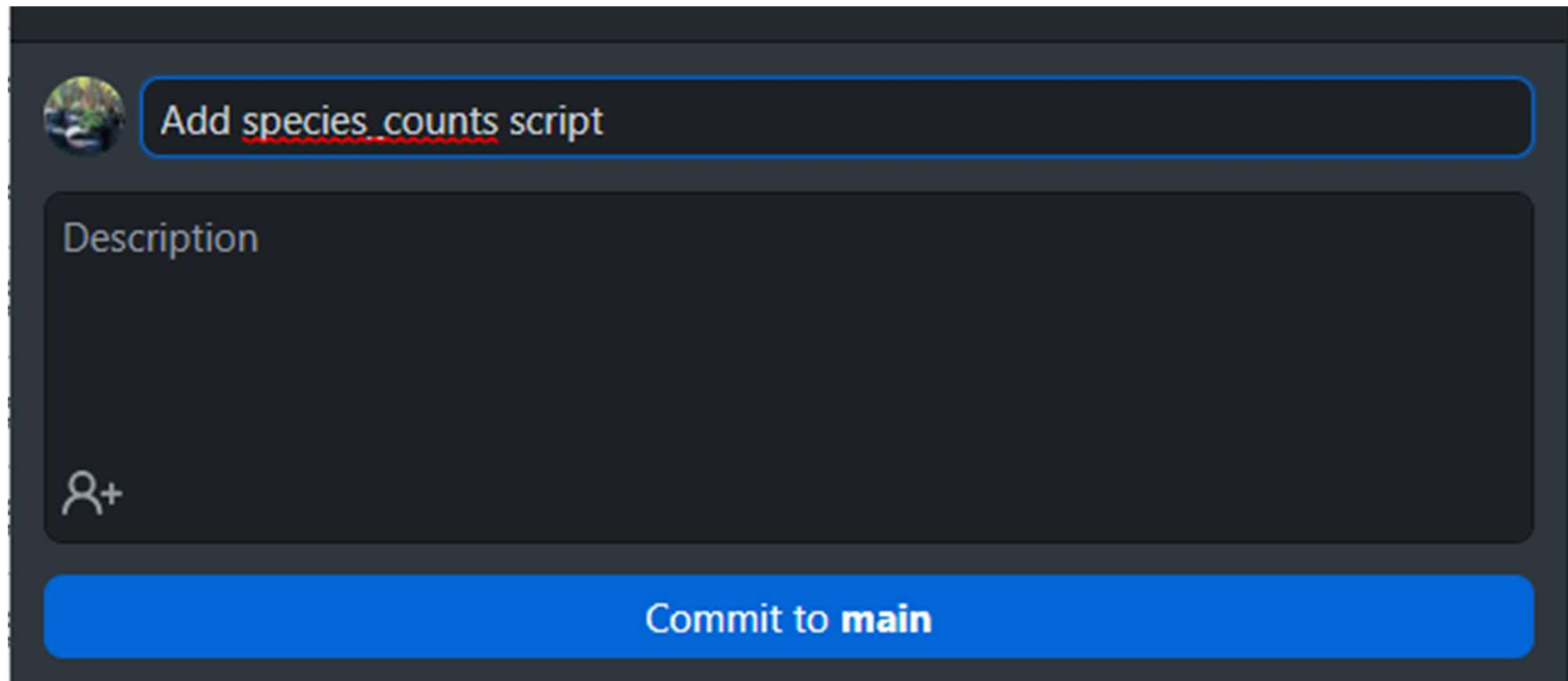
Basic idea - staging and committing



After Staging (selecting) changes, Commit changes to .git

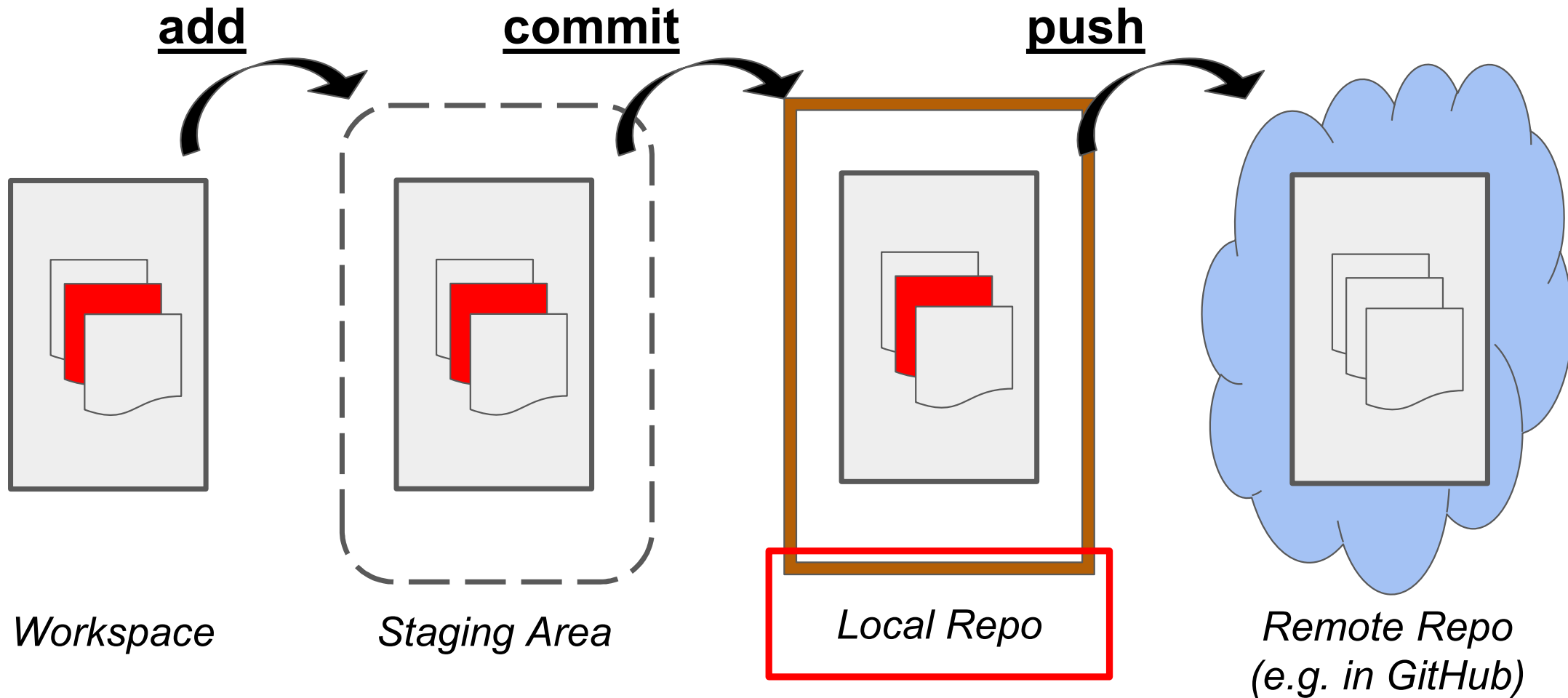


Write an informative 'Commit Message' and then commit

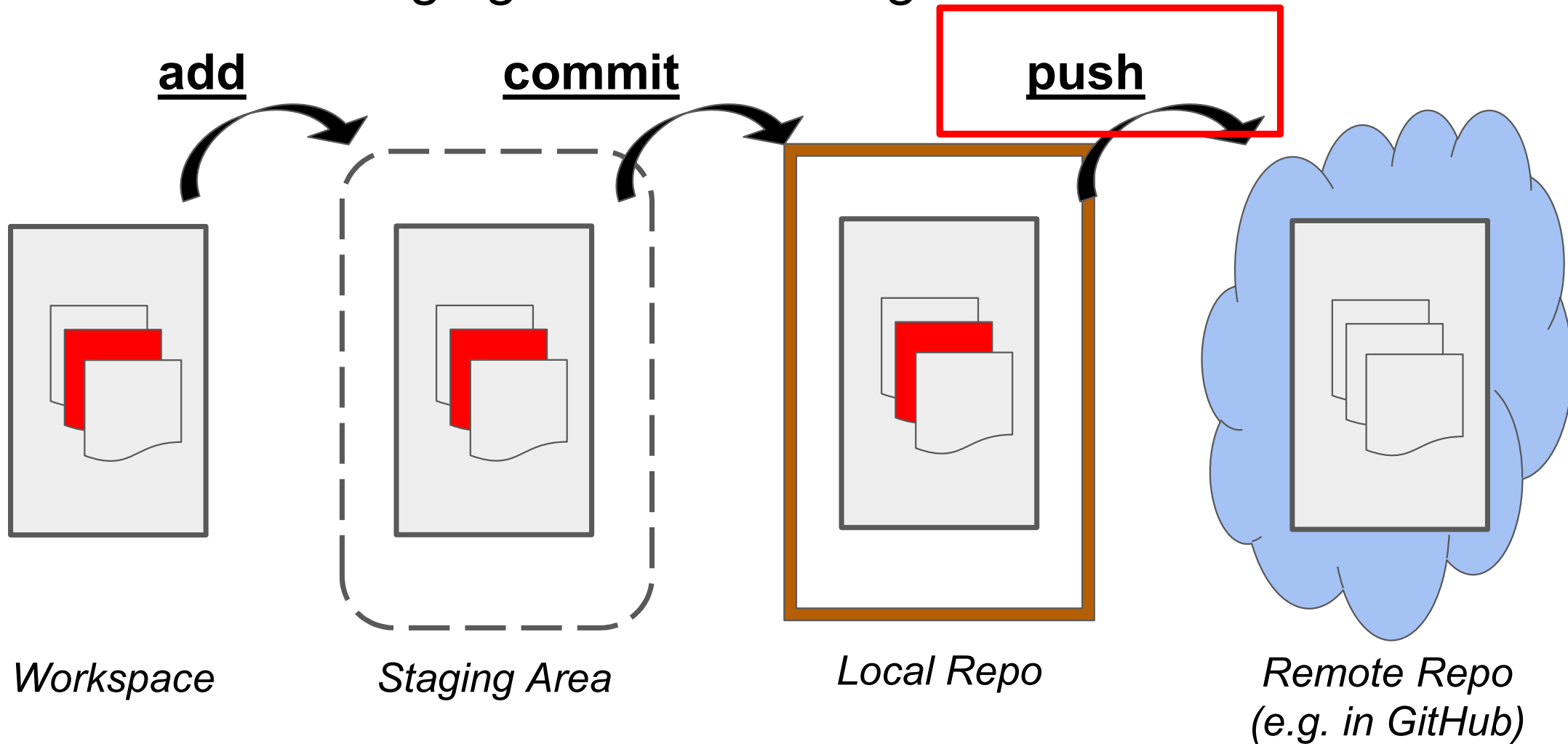


A screenshot of a commit message form in a dark-themed IDE. The form has a dark gray background with a blue border. At the top left is a circular profile picture of a person. To its right is a text input field containing the text "Add species_counts script", where "species_counts" is underlined in red. Below this is a large text area for the commit message, with the placeholder text "Description" in a light gray font. In the bottom left corner of the text area is a small icon of a person with a plus sign. At the bottom of the form is a prominent blue button with the text "Commit to **main**".

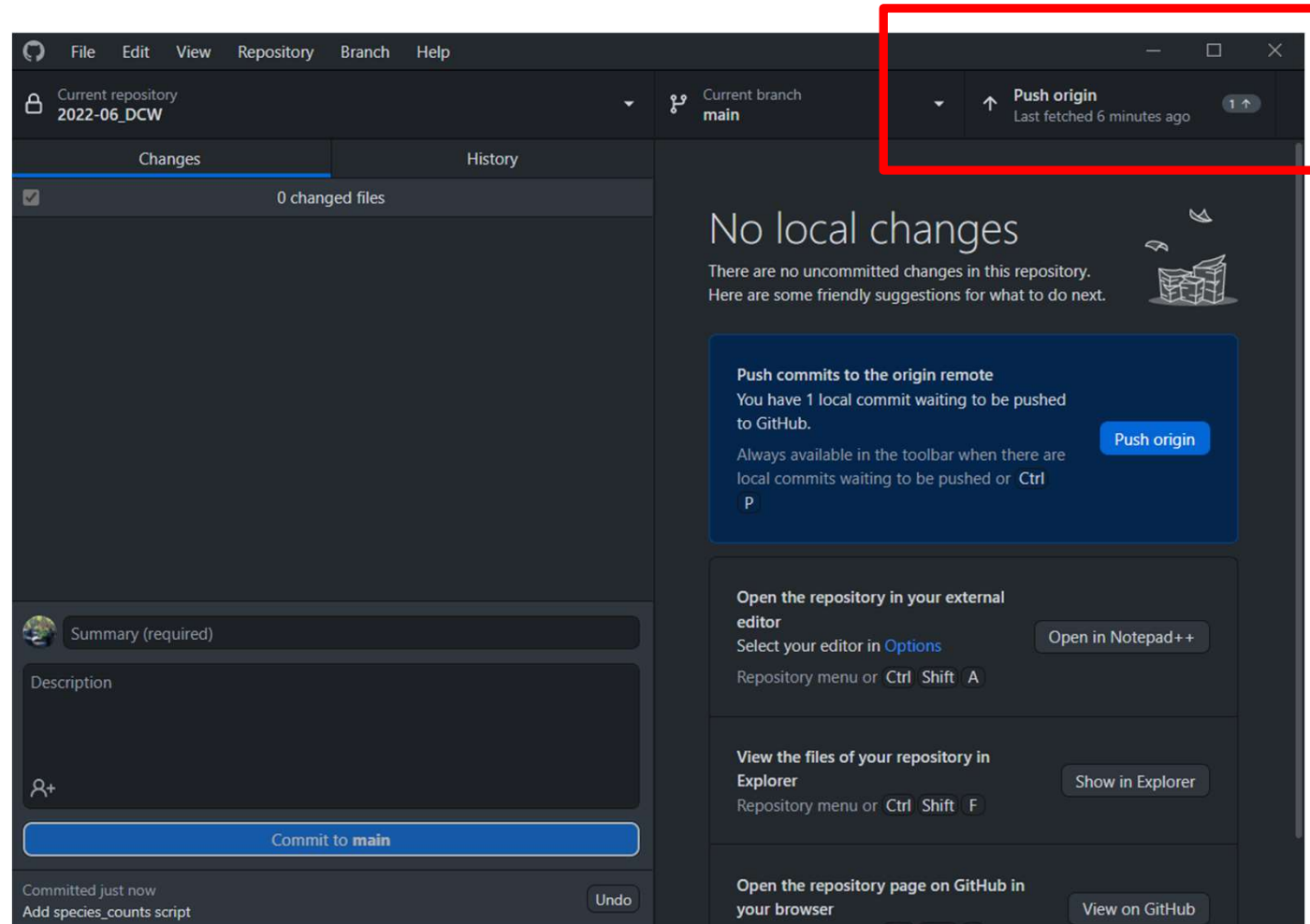
Basic idea - staging and committing



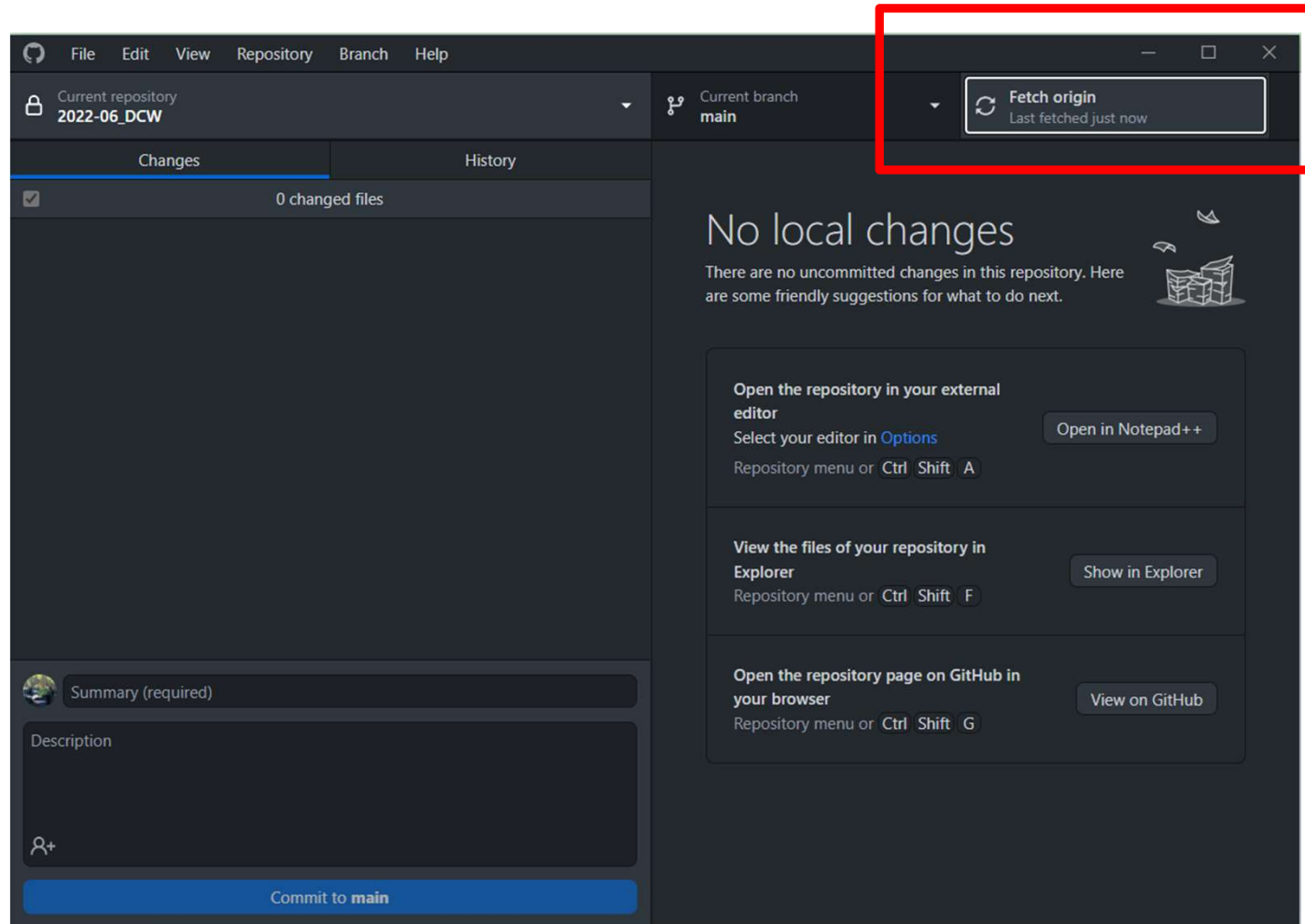
Basic idea - staging and committing



To send your local changes to the cloud, PUSH!

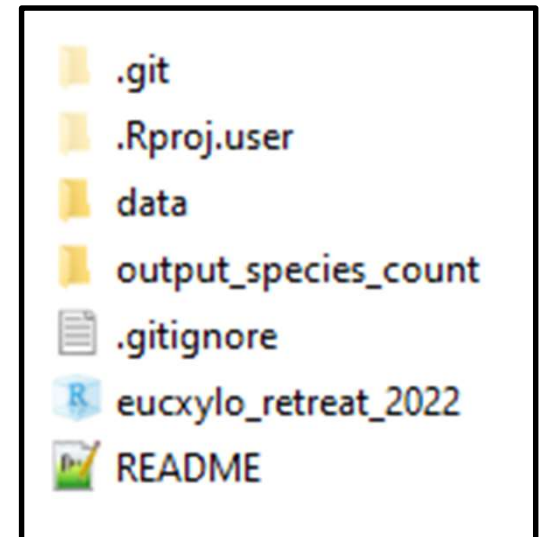


‘Fetch’ checks GitHub (‘origin’ repo) for changes...



EXERCISE: Add data to Repo (okay for small examples...)

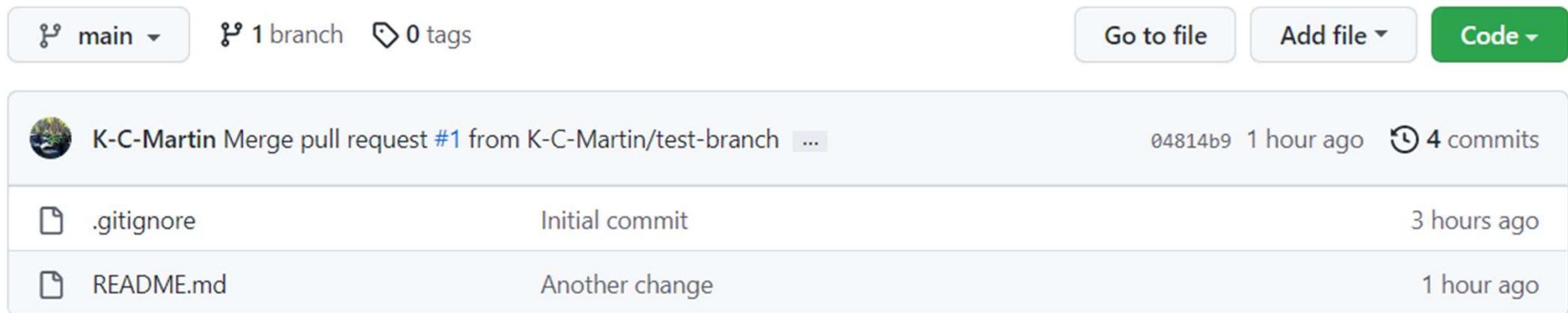
- Take the provided 'data' folder and copy it into your repo folder
- **Commit** and **Push** the changes to your repo using GitHub Desktop
- Inspect the data in GitHub (website)



EXERCISE: Commit changes to R Script

- In RStudio, paste the provided text into the `species_counts.R` file, and save
- Select all (Ctrl-A) and run the script (Ctrl-Enter)
- **Commit** and **Push** the changes to your repo using GitHub Desktop
- Navigate inside the 'output_species_count' folder in GitHub (website), and inspect the png and csv files

Look at the commit history in GitHub Desktop



The screenshot shows the GitHub Desktop interface. At the top, there's a header bar with a branch selector showing 'main', a branch count of '1 branch', and a tag count of '0 tags'. To the right are buttons for 'Go to file', 'Add file', and a green 'Code' button. Below the header, a commit history table is displayed. The first row shows a merge pull request by 'K-C-Martin' from 'test-branch' with commit hash '04814b9' and '4 commits' made '1 hour ago'. The subsequent rows show individual file commits: '.gitignore' with the message 'Initial commit' made '3 hours ago', and 'README.md' with the message 'Another change' made '1 hour ago'.

File	Commit Message	Time
Summary		
K-C-Martin Merge pull request #1 from K-C-Martin/test-branch		04814b9 1 hour ago 4 commits
.gitignore	Initial commit	3 hours ago
README.md	Another change	1 hour ago

You can also look at the 'History' tab in GitHub Desktop for the commits in the local Repo

You can inspect changes in each commit - note the value of 'informative commit messages'!

EXERCISE: What happens with conflicting edits?

- On GitHub (website) add some text to the README - **commit** the change with the message '*online edit*'
- On your computer, open the README in your local repo and add some different text - **commit** the change with the message '*local edit*'
- In GitHub Desktop, try to Push the change... you will be asked to Fetch online changes first... Fetch, and then try to Pull again

Resolving merge conflicts (remove the <<< >>> lines too!)

```
6
7 <<<<<< HEAD
8 A change made in the local repo in GitHub Desktop
9 =====
10 A change made in the online ('origin') repo on GitHub
11 >>>>>> 0bcba628bfff1bb6fa843e04fe228c170e85d7789
12
13
14
```

```
6
7
8 A change made by combining the edits previously made in the local repo in
  GitHub Desktop, AND in the online ('origin') repo on GitHub
9
10
11 Save the README file after you've resolved the conflict, and 'Continue merge'
12 ... then Push
13
```

Something cool in GitHub... Insights > Network

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

[Pulse](#)
[Contributors](#)
[Community](#)
[Traffic](#)
[Commits](#)
[Code frequency](#)
[Dependency graph](#)
[Network](#)
[Forks](#)

Network graph

Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.

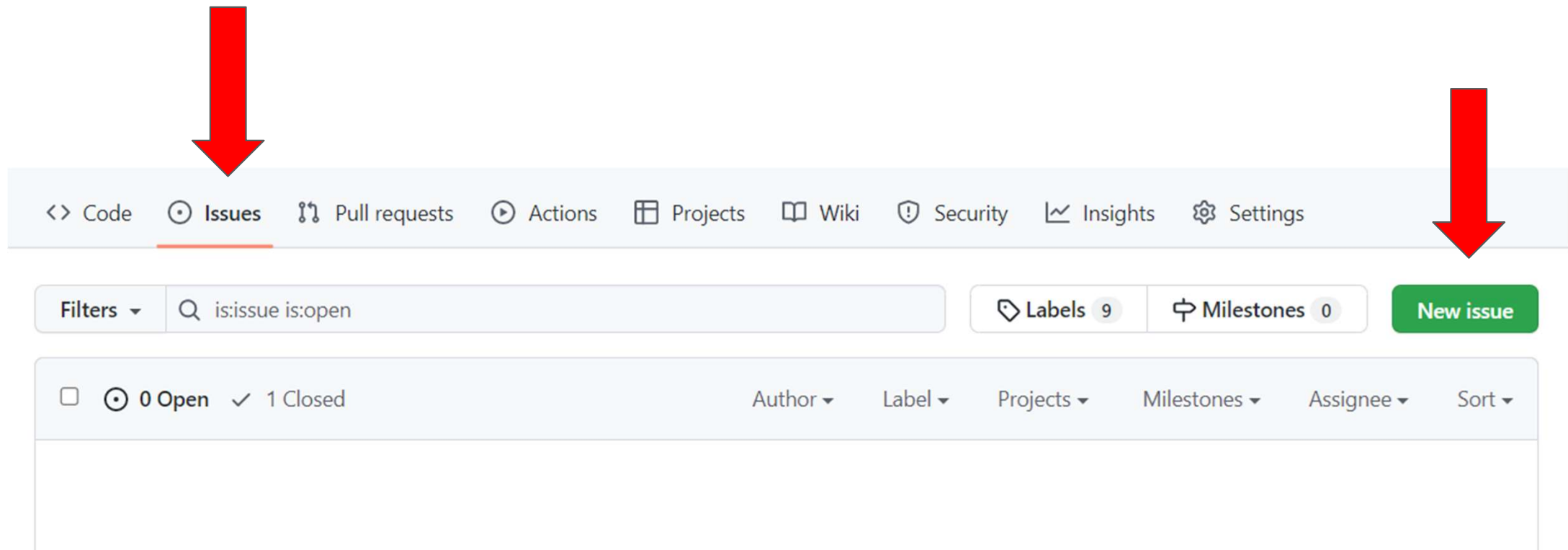
Owners	Jun
	29
K-C-Martin	30

TRY CLICKING ON THE 'COMMIT' DOTS

EXERCISE: Update README with GitHub Markdown

- In GitHub (website), paste the provided text into the README
- Look at the 'Preview' tab before you Commit (Google 'GitHub Markdown' later!)
- Commit the changes
- Click on the plot, look for the quickest way to the folder that contains its script

‘Issues’ are a way to track problems / tasks / ideas...



You can have long conversations with collaborators...

You can 'Open an Issue' directly from a line of code...



EXERCISE:

- Paste the link to your repo in the Teams Chat
- Visit the link to the repo that was posted right after you posted yours
- Navigate to the R Script in the subfolder (you can click on the plot in the README, and navigate from there)
- Open an issue on the line of code that creates the plot - suggest they make it grey-scale (not all journals accept colour figures... or it might cost more!!)

EXERCISE:

- Once you've had some feedback (via an Issue)... try to edit your code to respond to the suggestion ()
- Once your code works the way you want it to (and you've saved it), Commit and Push, and have a look at GitHub

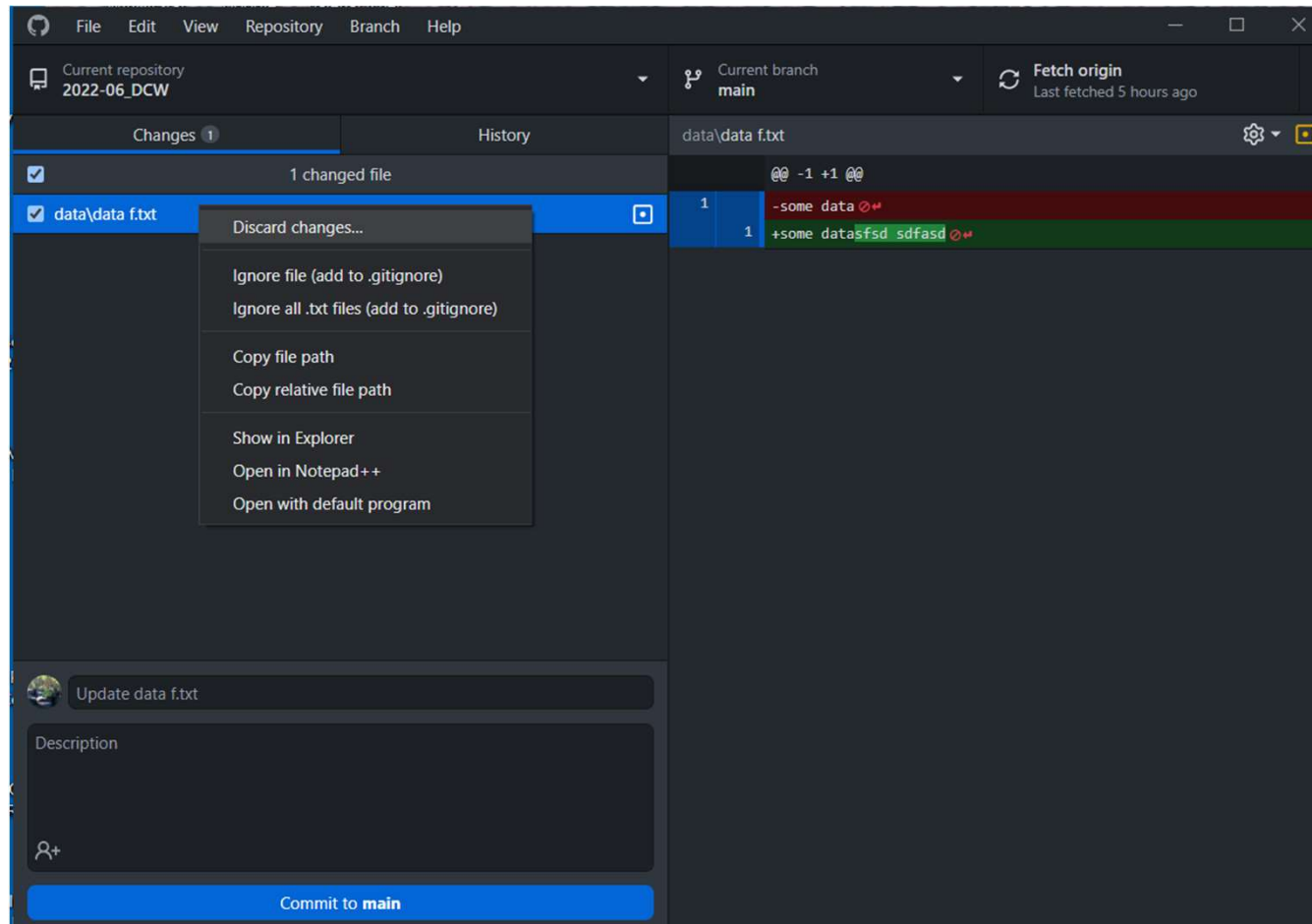
tip... try adding `+ scale_fill_grey()` after `geom_bar(stat = 'identity')` (line 77)...

pro-tip... try adding 'Closes #x' in the commit Description box... where 'x' is the issue number

EXERCISE: .gitignore makes Git Ignore things...

- Create a folder called 'notes' in your repo, and put something inside it (an empty txt file)
- Look at GitHub Desktop - you should see the new file in the data folder
- Open the .gitignore file and insert this at the top, then save: **notes/**
- Look at GitHub Desktop again...

You can 'discard changes' instead of committing



Right click on the staged files you want to discard changes in.

'Discard changes'

The file will go back to the way it was when you last committed it.

Turning your repo into a website

<> Code Issues Pull requests Actions Projects Wiki Security Insights **Settings**

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Actions

Webhooks

Environments

Pages

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Build and deployment

Source

Deploy from a branch

Branch

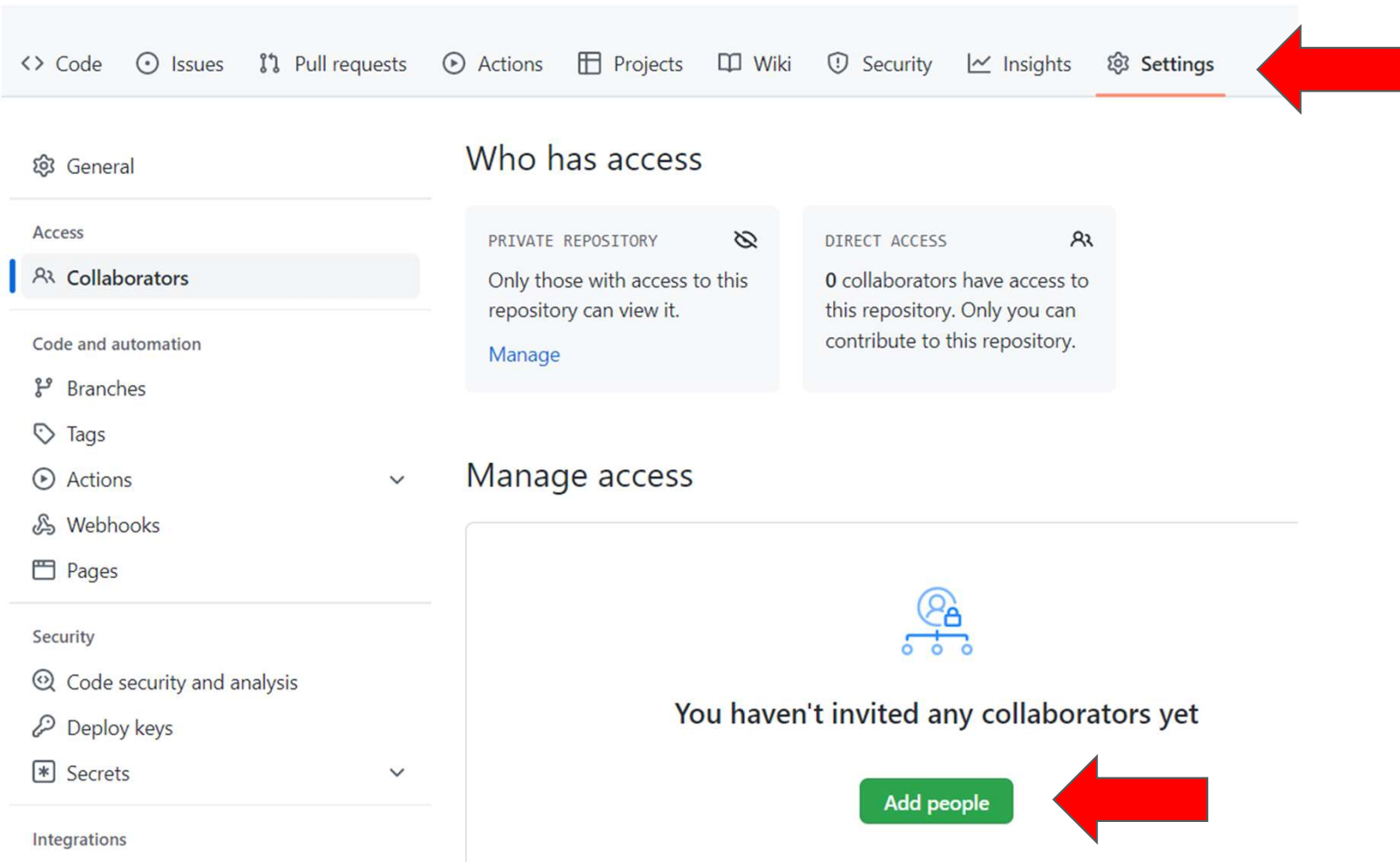
Your GitHub Pages site is currently being built from the `main` branch. [Learn more.](#)

main / (root) Save

Learn how to [add a Jekyll theme](#) to your site.

CHECK BACK LATER...

Adding collaborators (giving edit access to your repo)



The screenshot displays the GitHub repository settings interface. At the top, a navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. A red arrow points to the 'Settings' link. On the left sidebar, the 'Collaborators' tab is selected, indicated by a blue bar and a red arrow. The main content area is titled 'Who has access' and shows two access types: 'PRIVATE REPOSITORY' (with a lock icon) and 'DIRECT ACCESS' (with an open lock icon). The 'PRIVATE REPOSITORY' section states 'Only those with access to this repository can view it.' and includes a 'Manage' link. The 'DIRECT ACCESS' section states '0 collaborators have access to this repository. Only you can contribute to this repository.' Below this, the 'Manage access' section features a lock icon and the text 'You haven't invited any collaborators yet'. A green 'Add people' button is located at the bottom right, with a red arrow pointing to it.

<> Code Issues Pull requests Actions Projects Wiki Security Insights **Settings**

General

Access

Collaborators

Code and automation

Branches

Tags

Actions

Webhooks

Pages

Security


Code security and analysis

Deploy keys

Secrets


Integrations

Who has access

PRIVATE REPOSITORY 


Only those with access to this repository can view it.

[Manage](#)

DIRECT ACCESS 

0 collaborators have access to this repository. Only you can contribute to this repository.

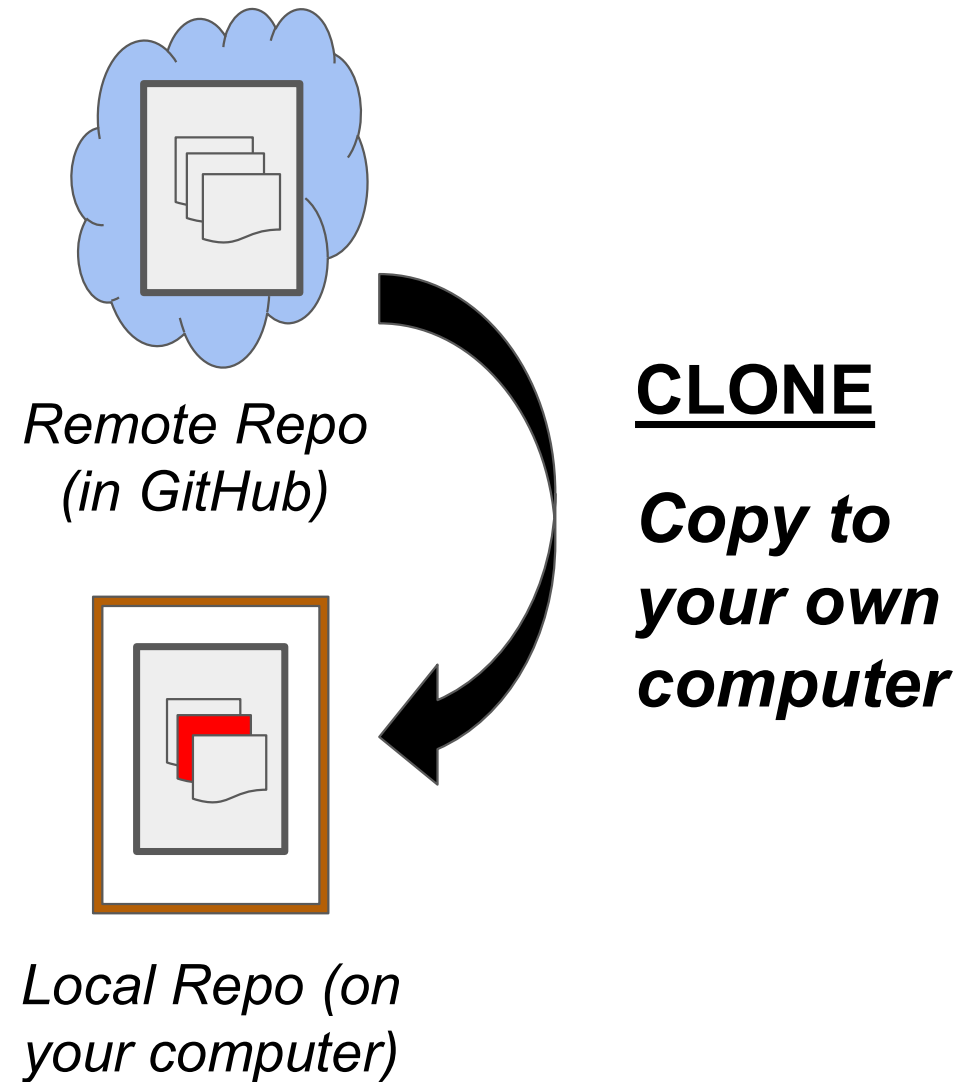
Manage access



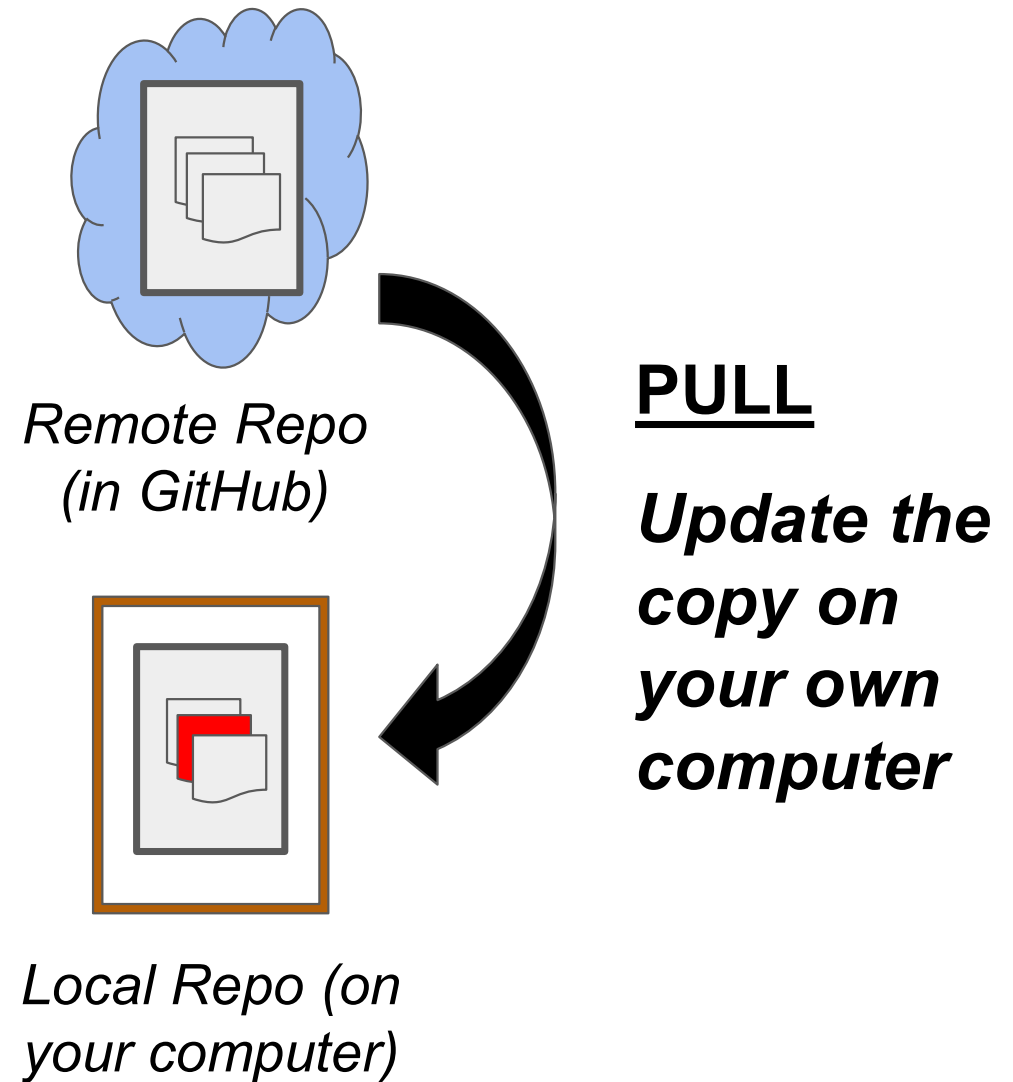
You haven't invited any collaborators yet

[Add people](#)

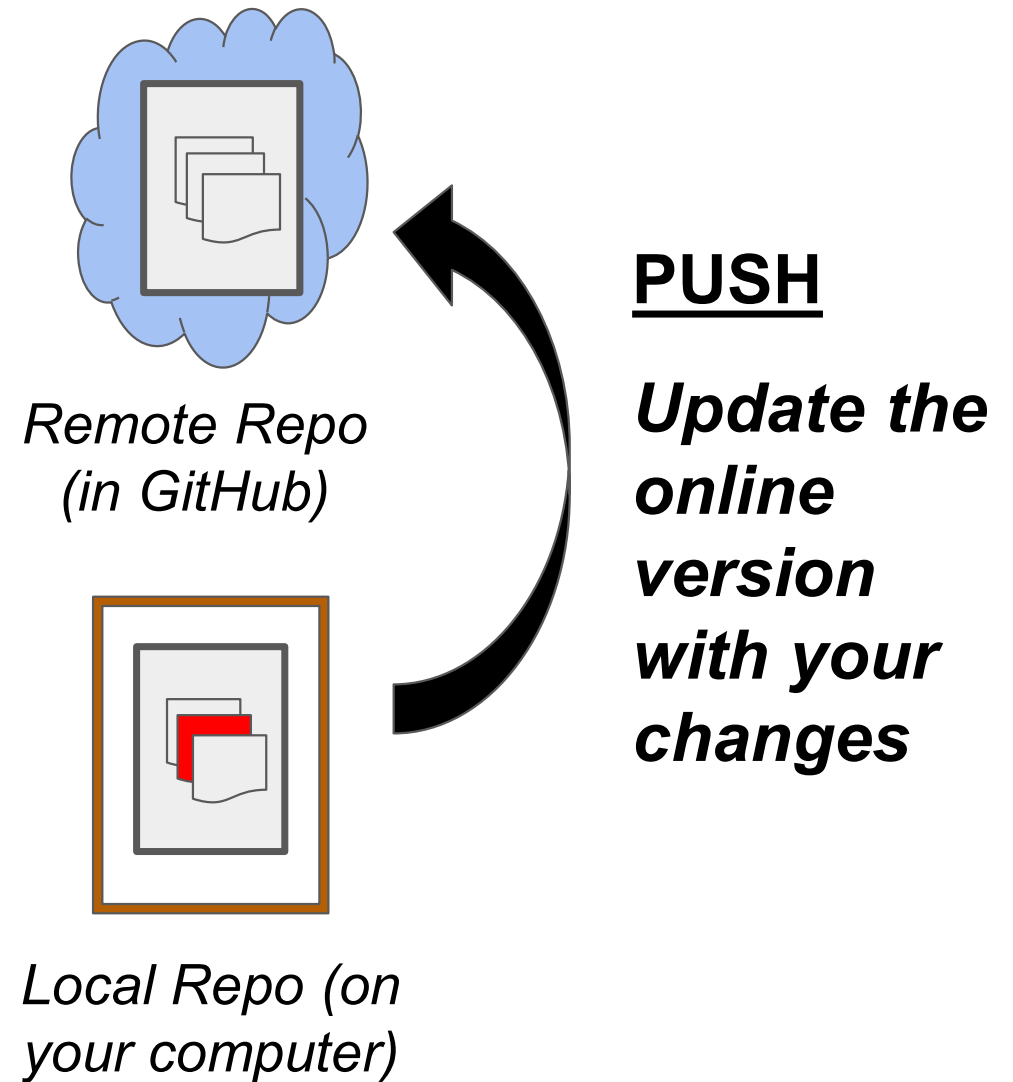
Cloning a repo



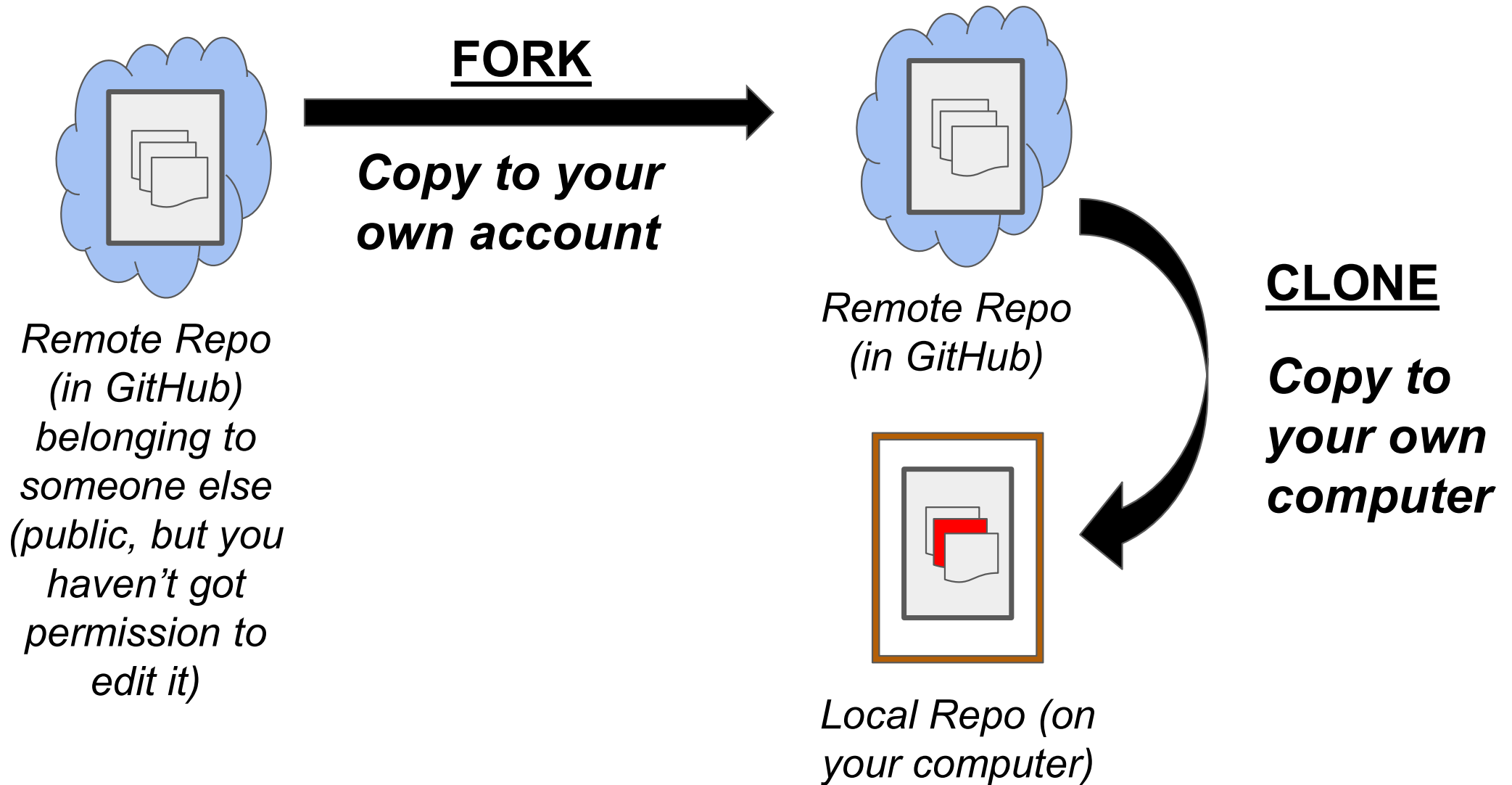
Pulling changes to your computer



Pushing changes back to GitHub



Cloning vs forking



Suggesting changes with a Pull Request (PR)

