



Intro to Git and GitHub



Center for Research Data
& Digital Scholarship

**How do you plan on
using Git and GitHub?**

Git vs GitHub

- Git: version control system
 - the actual software
- GitHub
 - cloud-based storage website



What is Git?

- Git is version control software, created by Linus Torvalds, the same person who created the Linux operating system.
- Git monitors files on your computer and tracks changes made to them over time
- Uses the command line

Why is it called “Git”?

- Linus Torvalds: "I'm an egotistical bastard, and I name all my projects after myself. First 'Linux', now 'Git'".
- Git: “A silly, incompetent, stupid or annoying person (usually a man).” (Wikitionary)

git / README 



Linus Torvalds Initial revision of "git", the information manager from hell

Code

Blame

168 lines (135 loc) · 8.2 KB

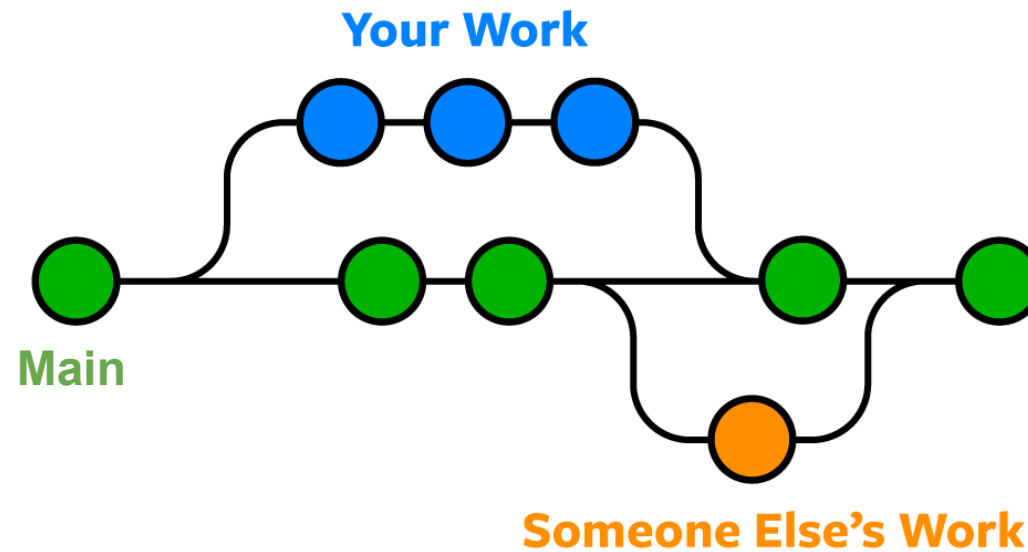
```
1
2     GIT - the stupid content tracker
3
4     "git" can mean anything, depending on your mood.
5
6     - random three-letter combination that is pronounceable, and not
7       actually used by any common UNIX command. The fact that it is a
8       mispronunciation of "get" may or may not be relevant.
9     - stupid. contemptible and despicable. simple. Take your pick from the
10      dictionary of slang.
11     - "global information tracker": you're in a good mood, and it actually
12       works for you. Angels sing, and a light suddenly fills the room.
13     - "goddamn idiotic truckload of sh*t": when it breaks
14
```

What is version control?

- Version control is the practice of tracking and managing changes to files.
 - Revert to various states of files
 - Modify items without harming the original copy
- Not limited to code
 - Can be used for documents, images, etc.

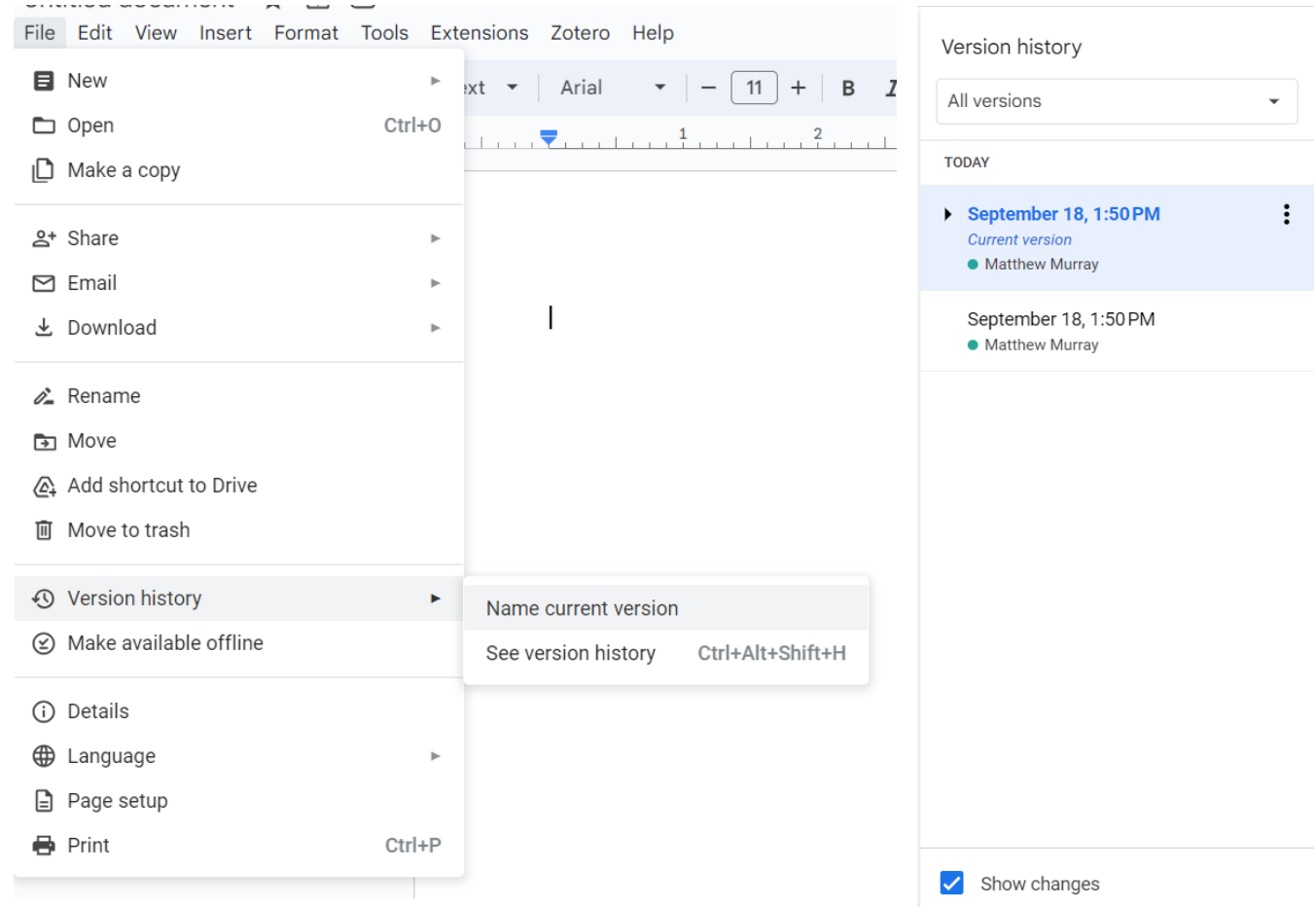
Benefits of version control

- Using version control provides
 - Clear tracking of the repo's history
 - Management and view of different branches
 - Collaboration through merging of branches

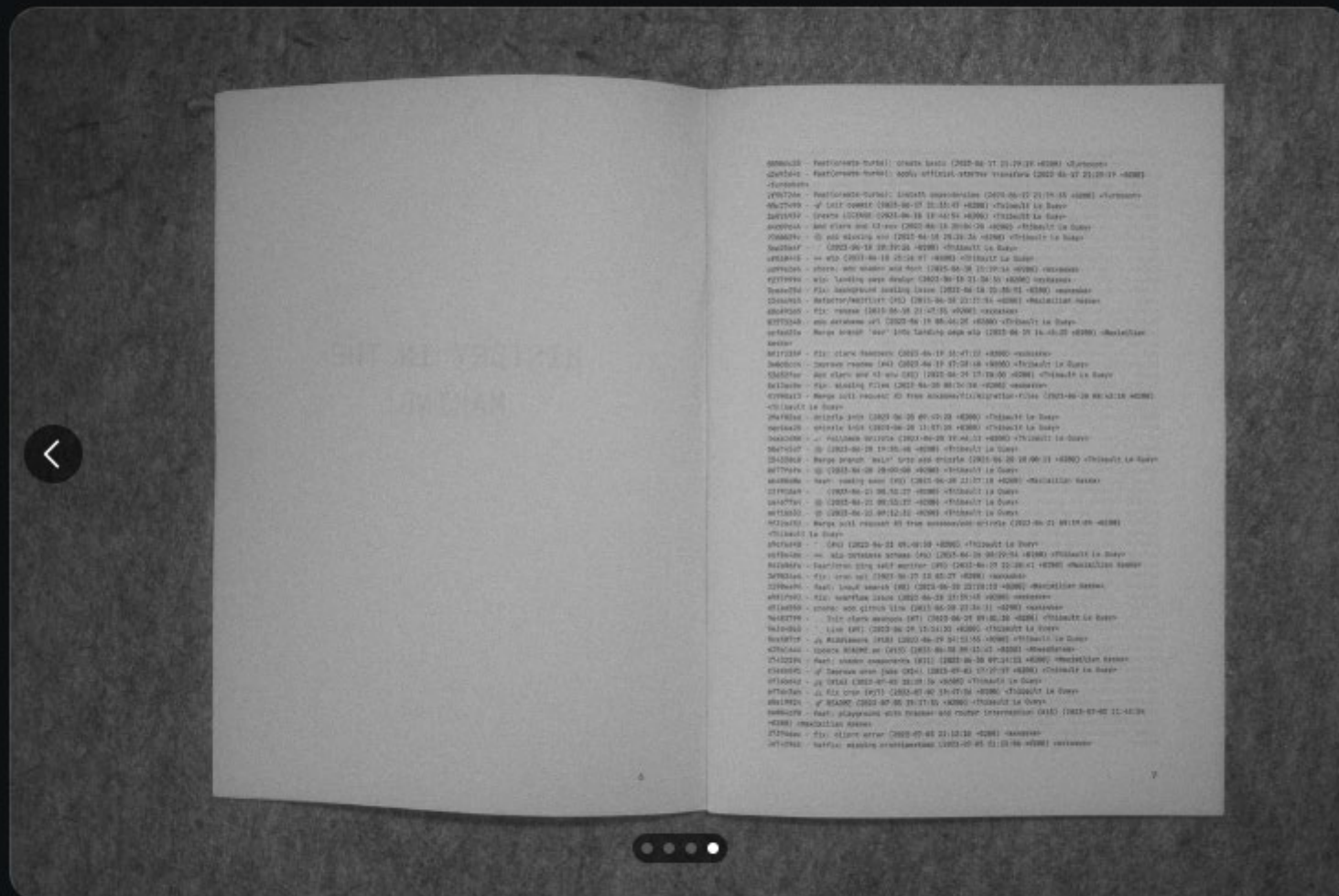


What is version control?

- Google Docs includes "Version History"
- This allows you to see what changes were made, when those changes happened, and who made them
- You can also revert to a previous version of your file



We made a zine from our Git history



We are an open-source project, so we decided to print our Git history to celebrate our 2nd anniversary.

https://www.reddit.com/r/zines/comments/1miy3bs/we_made_a_zine_from_our_git_history/

What is GitHub?

- GitHub is a Microsoft subsidiary that offers cloud hosting for Git repositories
- Provides a GUI (Graphical User Interface) for Git
 - Most (but not all) features are available in the GitHub Desktop client
- Allows for easy collaboration and sharing
 - Issue queues for bugs and features, pull requests, and more
- GitHub basic is free (has limits)
 - Hosts both open and private repositories
- GitHub Enterprise (free for CU affiliates)
 - Includes cloud-based development environments
 - <https://oit.colorado.edu/services/business-services/github-enterprise>

What Can I Use Git and GitHub For?

Git

- Track and manage changes of files
- Document changes
- Collaborate with others

GitHub

- Off-site backup for Git repositories
- Contribute to Open Source projects
- Communicate with team members
- Communicate with your users
- Join communities
- Bug tracking
- Website hosting

CU-Boulder-CRDDS / Open-Access-Odyssey

🔍 Type / to search

<> Code

🔍 Issues 1

🔗 Pull requests

🎬 Actions

📁 Projects

🛡 Security

📈 Insights

⚙ Settings

Open-Access-Odyssey

Public

🌟 Edit Pins

👁 Watch 0

🍴 Fork 0

★ Star 0

🔗 main

🔗 2 Branches

🏷 0 Tags

🔍 Go to file

Add file

<> Code

⚙

👤 ellery-galvin

Add ID to ring image and update CSS with target

✓

fbfd88b · 3 weeks ago

🕒 37 Commits

📄 .gitignore	Adding .gitignore. Resizing images.	3 weeks ago
📄 CU-Scholar.png	Images and styles.	last month
📄 ILL.png	Images and styles.	last month
📄 LICENSE	Create LICENSE	3 weeks ago
📄 OA-Logo.jpeg	Images and styles.	last month
📄 OA_Week_2025.html	Add ID to ring image and update CSS with target	3 weeks ago
📄 OneSearch.png	Updating files, adding image	2 months ago
📄 README.md	Update README.md	3 weeks ago
📄 UnderGrad-01.png	Fixing issues.	3 weeks ago
📄 University_Libraries_left.png	Updating logo.	3 weeks ago
📄 grad-01.png	Fixing issues.	3 weeks ago

About

⚙

An interactive text game made in Twine for OA Week 2025.

🔗

cu-boulder-crdds.github.io/Open-Access...

📖

Readme

📜

CC-BY-SA-4.0 license

📈

Activity

📋

Custom properties

★

0 stars

👁

0 watching

🍴

0 forks

Report repository

Releases

No releases published

🔗

Create a new release

Packages



Try to navigate academic life without spending your takeout food budget!

Learn about:

- Access to scholarly articles
- Publishing scholarly articles
- Impact and scholarly monographs

Choose your character:

Undergrad Student

Grad Student

Faculty Member

Alternatives to...

- Git (version control)
 - Apache Subversion (SVN)
 - Mercurial SCM
 - CVS (Concurrent Versions System)
- GitHub (hosting)
 - Codeberg
 - GitLab
 - Forgejo
 - Bitbucket



When not to use GitHub

- When you are looking for long-term preservation
 - There's no guarantee Microsoft will keep GitHub around forever
 - Lots of source-code-hosting platforms no longer exist
 - Thousands of URLs in research articles no longer work as they point to code hosted by defunct services
- When you want your code to be cited
 - [Zenodo](#) is an Open Access repository that can be linked to GitHub
 - Allows you to archive a specific release of public GitHub projects
 - Creates a DOI for the archive that you can use in citations

Zenodo.org will be unavailable for 2 hours on September 29th from 06:00-08:00 UTC. See [announcement](#).

September 2, 2023

Software

Open Access

PyGMT: A Python interface for the Generic Mapping Tools

 Tian, Dongdong;  Uieda, Leonardo;  Leong, Wei Ji;  Schlitzer, William;  Fröhlich, Yvonne;  Grund, Michael;  Jones, Max;  Toney, Liam;  Yao, Jiayuan;  Magen, Yohai;  Jing-Hui, Tong;  Materna, Kathryn;  Belem, Andre;  Newton, Tyler;  Anant, Abhishek;  Ziebarth, Malte; Quinn, Jamie; Wessel, Paul

PyGMT is a library for processing geospatial and geophysical data and making publication quality maps and figures. It provides a Pythonic interface for the [Generic Mapping Tools \(GMT\)](#), a command-line program widely used in the Earth Sciences.

The development of PyGMT has been supported by NSF grants OCE-1558403 and EAR-1948603.

Preview

 baseline-images.zip

baseline-images

- test_basemap.png 6.2 kB
- test_basemap_compass.png 71.2 kB
- test_basemap_loglog.png 24.5 kB
- test_basemap_map_scale.png 31.0 kB
- test_basemap_polar.png 31.9 kB

18,263

 views

1,446

 downloads

[See more details...](#)

Available in

GitHub

Indexed in

OpenAIRE

Publication date:

Zenodo.org will be unavailable for 2 hours on September 2, 2023

September 2, 2023

PyGMT: A Python interactive Mapping Tools

 Tian, Dongdong;  Uieda, Leonardo;  Leong, Wei Ji Jones, Max;  Toney, Liam;  Yao, Jiayuan;  Magen, Newton, Tyler;  Anant, Abhishek;  Ziebarth, Malte; 

PyGMT is a library for processing geospatial and geophysical data. It provides a Pythonic interface for the [Generic Mapping Tools](#) (GMT) Sciences.

The development of PyGMT has been supported by N

Preview

 baseline-images.zip

 baseline-images

-  test_basemap.png
-  test_basemap_compass.png
-  test_basemap_loglog.png
-  test_basemap_map_scale.png
-  test_basemap_polar.png

Versions

Version v0.10.0 Sep 2, 2023

10.5281/zenodo.8303186

Version v0.9.0 Mar 31, 2023

10.5281/zenodo.7772533

Version v0.8.0 Dec 30, 2022

10.5281/zenodo.7481934

Version v0.7.0 Jul 1, 2022

10.5281/zenodo.6702566

Version v0.6.1 Apr 11, 2022

10.5281/zenodo.6426493

[View all 17 versions](#)

Cite all versions? You can cite all versions by using the DOI [10.5281/zenodo.3781524](#). This DOI represents all versions, and will always resolve to the latest one. [Read more.](#)

24.0 KB

31.0 kB

31.9 kB

18,263

 views

1,446

 downloads

[See more details...](#)

Available in

GitHub

Indexed in

OpenAIRE

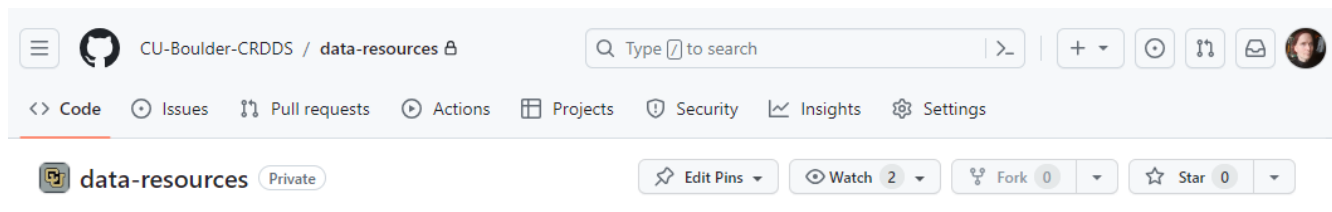
Publication date:

Ways to interact with GitHub

- The website
- GitHub Desktop
- Using Git and the command line

Website

- Create and manage projects
- Upload and download files
- Write documentation



midnitenibrary Updating README.md		3c43dbf · 4 months ago	🕒 3 Commits
📄 LICENSE	Initial commit	4 months ago	
📄 README.md	Updating README.md	4 months ago	
📄 dataset_README_template.txt	Adding README template	4 months ago	

[📖 README](#)
[📄 CC0-1.0 license](#)
[✎](#)

About

Data-related resources for researchers.

- Readme
- CC0-1.0 license
- Activity
- Custom properties
- 0 stars
- 2 watching
- 0 forks

data-resources CU-Boulder-CRDDS / data-resources 🔍 Type / to search

<> Code Issues Pull requests Actions Projects Security Insights Settings

Data-related resources for res

 [data-resources](#) / in [main](#)

Cancel changes [Commit changes...](#)

Edit Preview Spaces 2 Soft wrap

```

1 # data-resources
2 Data-related resources for researchers.
3
4 This is a place to find University of Colorado Boulder resources created for researchers.
5
6 * [dataset README template.txt](https://github.com/CU-Boulder-CRDDS/data-resources/blob/main/dataset_README_template.txt): This is a template of a
  README file created for researchers who want to deposit data into [CU Scholar](https://scholar.colorado.edu/), University of Colorado Boulder
  Libraries' institutional repository.
7

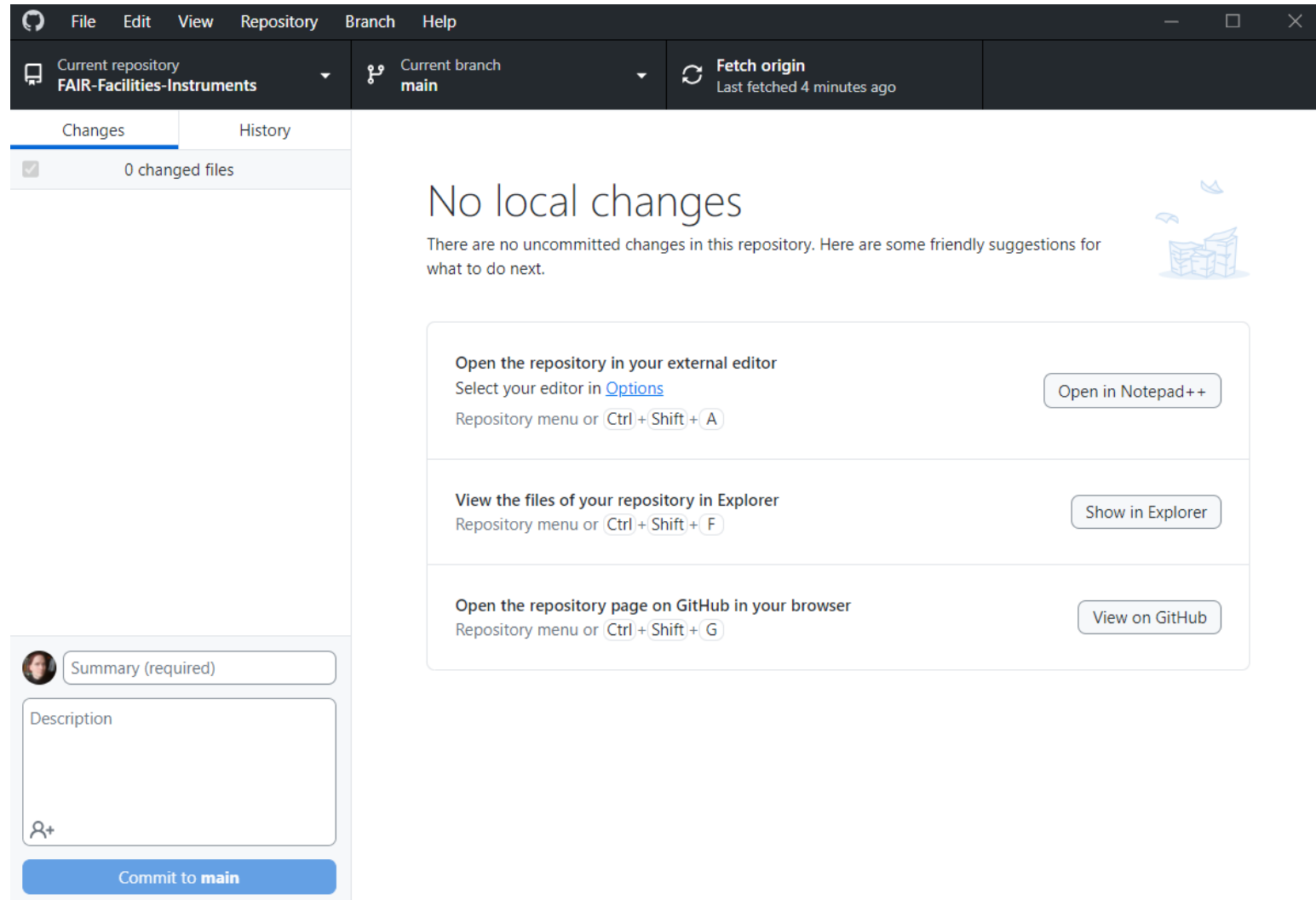
```

Use `Control + Shift + m` to toggle the `tab` key moving focus. Alternatively, use `esc` then `tab` to move to the next interactive element on the page.

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

GitHub Desktop

- Create, clone, and fork projects
- Commit changes and submit contributions
- Many people who use Git frequently prefer to use the command line
- Other graphical Git interfaces exist



Be careful where you put your GitHub directory

The screenshot shows the Microsoft OneDrive web interface for a user named Matthew Murray. The breadcrumb path is 'My files > UCB > Documents > GitHub'. Below this path, a table lists the contents of the 'GitHub' directory:

Name	Modified	Modified By	File size	Sharing	Activity
.git	September 12, ...	Matthew Murray	5 items	Private	
.github	April 29, 2024	Matthew Murray	2 items	Private	

- Microsoft OneDrive uploads a lot of stuff to the cloud, which you don't always want

Getting Started with Git

Definitions

- Trunk/Master: Main version of a repository
- Clone: A copy of a repository that is stored locally instead of online
- Branch: A parallel version of a repository that allows modifications to be made without affecting the main file
- Fork: A branch that is not intended to be merged back into the main file
- Pull/Fetch: Getting the latest version of a repository
- Merge: Combining changes from different branches/versions of files
- Commit: A saved snapshot of your project and any changes you have made
- Push: Sending your local changes to an online repository

Things to Remember about the Git Workflow

- Git isn't an automatic process
- In order to use it, you need to integrate Git into your daily workflow
- You set it up for a project, on a directory on your hard drive, and from that point forward Git will start watching for any changes in that directory.
- You do your work as normal, using all the same software and tools you're used to using
- At any point, you can ask Git to tell you what changes it has detected and it will show you a detailed report of every addition and deletion that it has detected

Things to Remember about the Git Workflow

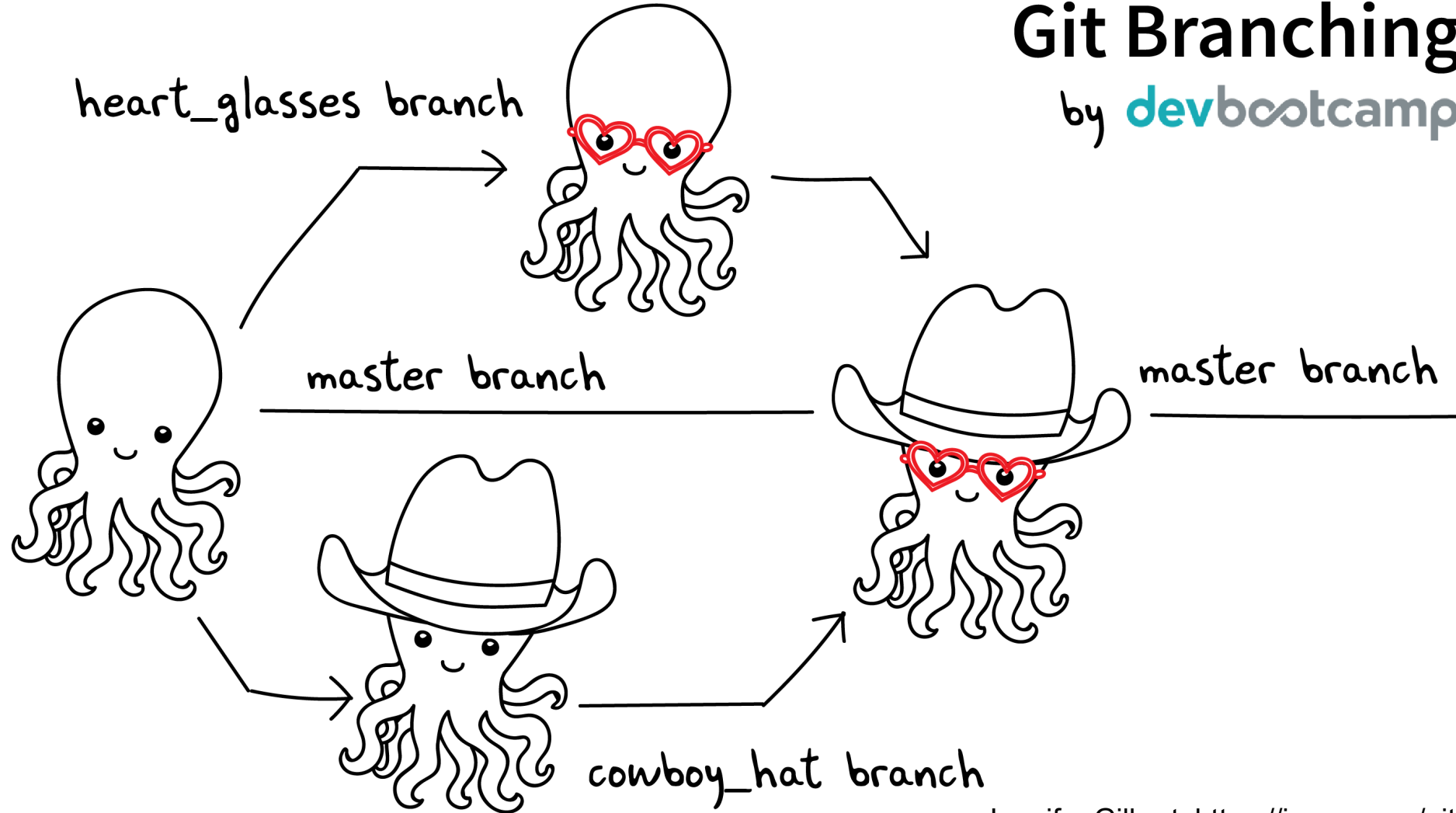
- Once you've finished working on something, it's time to make a commit, which you can think of as a snapshot of the state of your project. Each commit records all the changes that happened to get it from the previous version of your project to the current one.
- When you make a commit, you select some (or all) of the changes that git has detected in your directory, and you mark them for inclusion in your commit. Then, you provide a message to describe the changes that you've made and create your commit.
- The commit is added to a journal of sorts, where you can see all the various commits that you've recorded for your project.
- Then, at any point in the future, we can examine any of the commits, look at the files that changed, and the message that was created for the commit. If you decide that you need to, you can “rewind” your project to the state that's described by the commit.

Git Workflow - Branches

- Branches allow you to create multiple versions of your project, and switch between them seamlessly.
- Git is often described as a “distributed” version control system. When git is described as “distributed” it means that it’s designed to work with many copies of the same repository in many different locations, different computers or servers. Those individual copies can then speak to each other, to exchange information and share changes between each of them.
- If you’re working on your own, this might not seem like a useful feature, but it becomes really valuable when you’re working with other people.
- At any point, each person can start working independently on their own copy of code and use the branching and merging functionality to do their work.
- Each branch lives on a different computer entirely, but the changes can be pushed back into the main copy.

Branching & Merging

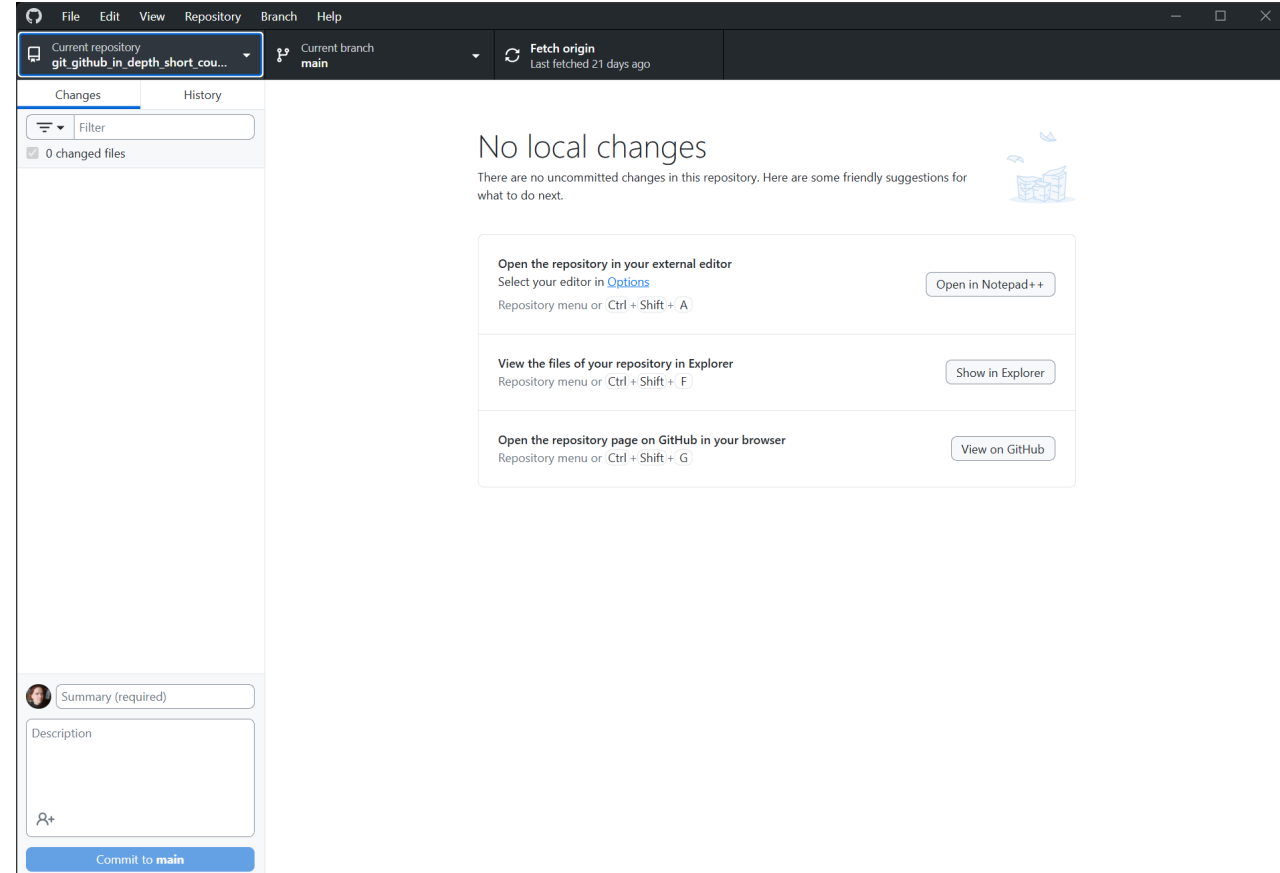
Git Branching
by **dev**bootcamp



Interacting with Git & GitHub

GitHub Desktop

- Easiest way to get started
- Offers a very streamlined experience
- Simplicity comes at the expense of power and flexibility. A lot of advanced Git features aren't available in the GitHub Desktop client.
- GitHub desktop only works with local Git and GitHub. If your team is using GitLab, BitBucket, or some other cloud repository hosting service, this client will not speak to those.



The Command Line

- The most powerful, and the best way to really learn the ins and outs of Git.
- Has the steepest learning curve. The command line interface requires you to learn and memorize all the available commands, the workflow, etc.
- Not a good option for beginners unless you're the kind of person that's already used to working on the command line.

Getting Git on your local machine

- Many systems have Git installed; however, you may need to download it on your local machine:
- See <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git> for more information on installing Git
- You don't need to do this if you're just using the graphic version of GitHub

Linking Git to GitHub

- Once downloaded, we can then configure Git with our GitHub username and email via the command line. This allows us to interact with our GitHub more easily.

- First, set your username. For example, if my GitHub username is gh-user, then I would do the following:

```
$ git config --global user.name "gh-user"
```

- Now, set your email. For example, if my email for GitHub is gh-user@gmail.com, then I would do the following:

```
$ git config --global user.email "gh-user@gmail.com"
```

- Confirm Git has been configured (should show your entered info)

```
$ git config --list
```

Personal Access Token

- Allows you to verify your identity with GitHub
- For more information, see <https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/managing-your-personal-access-tokens>
- Downside is that you need to enter your username and then the Personal Access Token as your password for events such as:
 - Any interaction with a private repo
 - Pushing to a public repo
- There are ways to store your username and token, but these require third-party software

SSH Keys

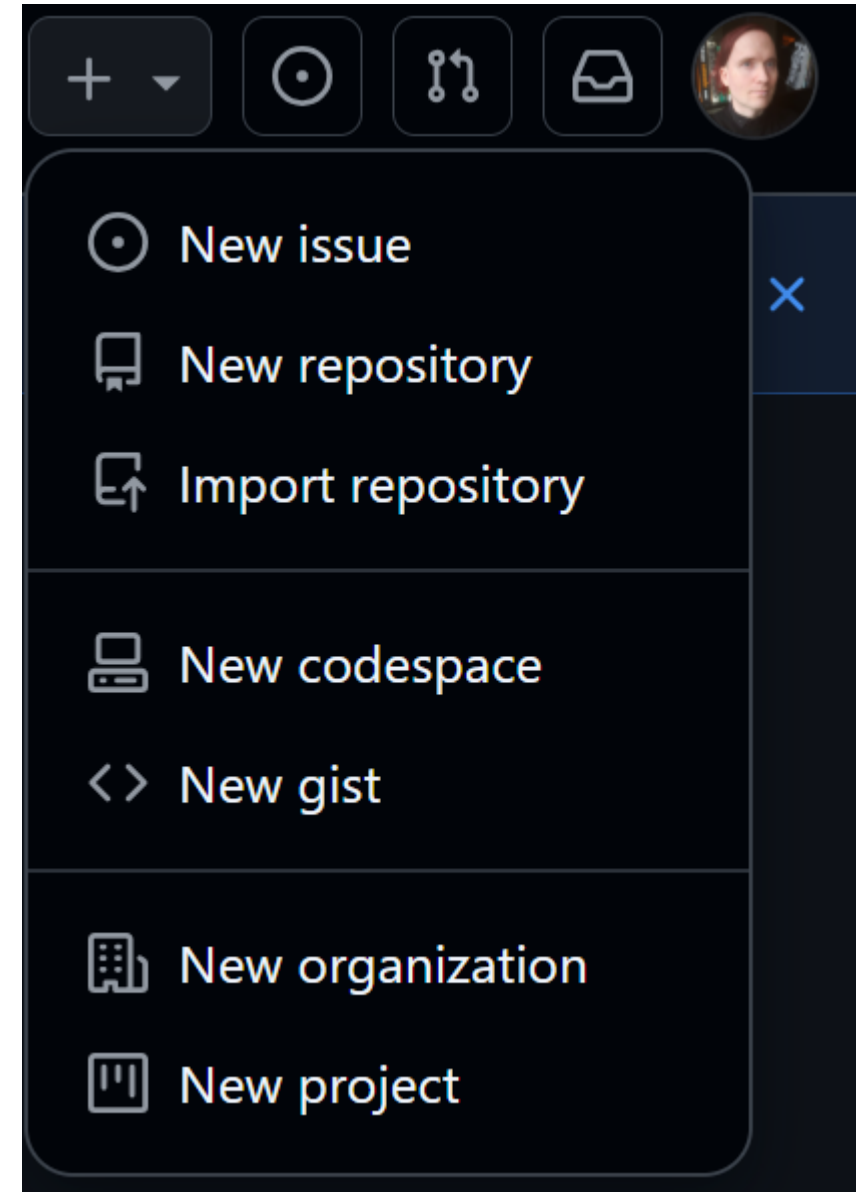
- Alternative way to verify your identity with GitHub
- For more information see:
<https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>
 - Be sure to select the proper Operating System when using the link
- Setup is more involved, but makes it so that you never have to enter your username and token when interacting with a private repo or pushing to a public repo

Setting up a new GitHub repo

New GitHub Repos

When you first create a repo locally, you will need to setup a new repository on GitHub too

- Go to <https://github.com>
- Sign in
- Click on “Create New Repository” or just “New”



Creating a repo

- Choose an “owner”
- Add a description
- Choose visibility
- Add a README
- Add a .gitignore
- Add a license

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).
Required fields are marked with an asterisk (*).

1 General

Owner *

Choose an owner ▾

Repository name *

Great repository names are short and memorable. How about **turbo-octo-adventure**?

Description

0 / 350 characters

2 Configuration

Choose visibility *

Choose who can see and commit to this repository

Public ▾

Start with a template

Templates pre-configure your repository with files.

No template ▾

Add README

READMEs can be used as longer descriptions. [About READMEs](#)

Off ☐

Add .gitignore

.gitignore tells git which files not to track. [About ignoring files](#)

No .gitignore ▾

Add license

Licenses explain how others can use your code. [About licenses](#)

No license ▾

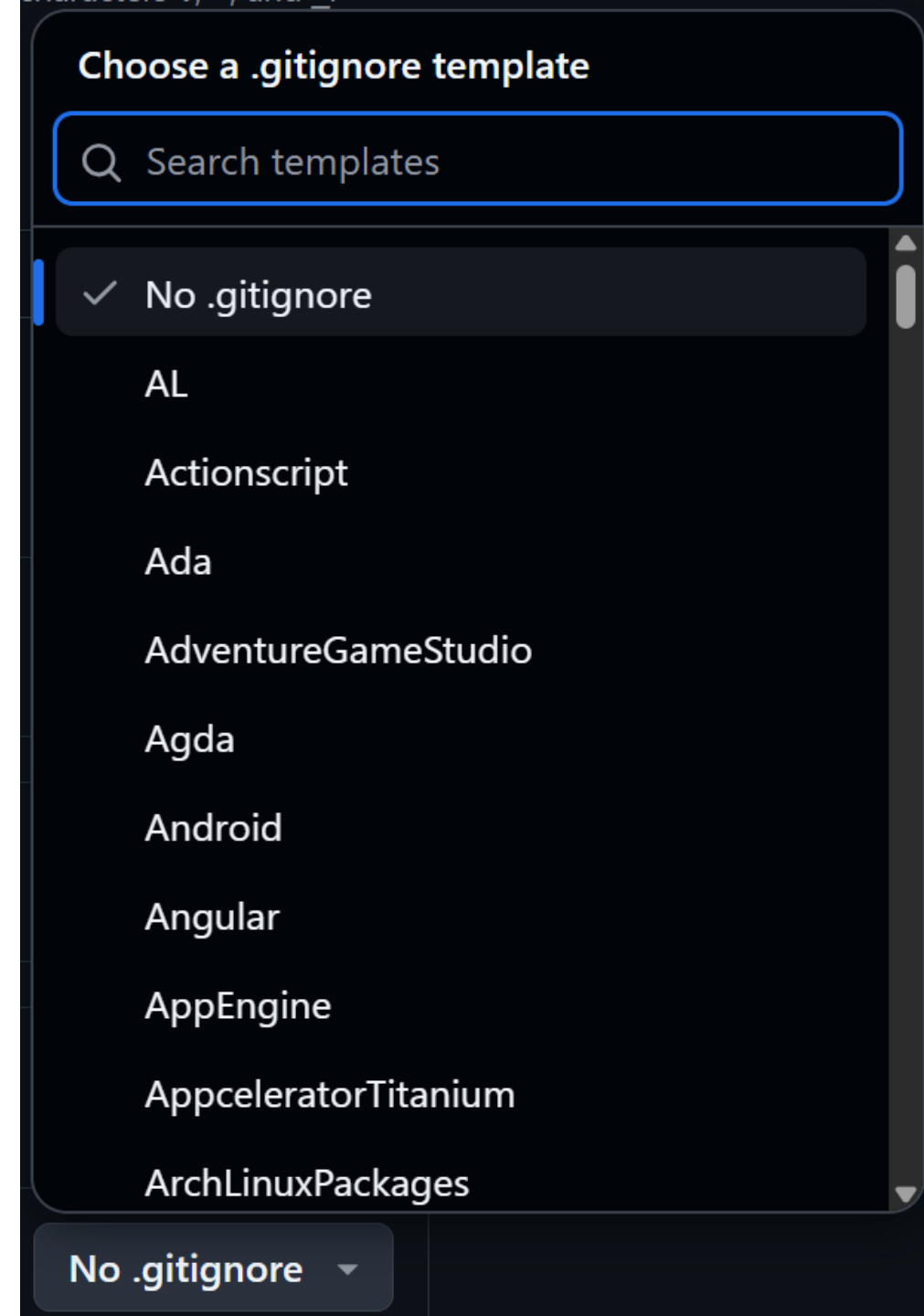
Create repository

README Files & Documentation

- Include documentation with your project in GitHub so that others (and you) know what your project is and how it should be used.
- A README.md (markdown) file can be included in your GitHub project and will display on the front page
- What to include in a README:
 - What your project does
 - How people can use it
 - Who you are and how to contact you
 - License information
- Lots of examples and templates available
- We can provide feedback on documentation

.gitignore

- Specifies what items (files, directories, etc.) should never be tracked by Git
- Use for information that needs to be kept private or is irrelevant to a project
- Use a template or create your own



Example .gitignore

- [CHAFF](#) (Concealed, Hidden, And Forgotten Files)
- .DS_Store files are created automatically by MacOS computers and are not visible to those using that OS
- These files are used to keep track of metadata used by Finder—such as display options or icon positions—and Spotlight's search features

```
1
2 # DS store created on a mac
3 .DS_Store
4 ._DS_Store
5 **/.DS_Store
6 **/._DS_Store
7
8 # PowerPoint temporary
9 ~$*.ppt*
10
```

Licenses

- A license is a legal permission that specifies what you are (and are not) allowed to do a work
- Applies to how you can use, modify, and distribute the work
- Apply to creative work, software, datasets, intellectual property, and more
- Consider how permissive do you want to be
- For code, check out the website “Choose an open source license”
<https://choosealicense.com/>
- Feel free to reach out and ask us
- Note: I am not a lawyer

Choose a license

✓ No license

Apache License 2.0

GNU General Public License v3.0

MIT License

BSD 2-Clause "Simplified" License

BSD 3-Clause "New" or "Revised" License

Boost Software License 1.0

Creative Commons Zero v1.0 Universal

Eclipse Public License 2.0

GNU Affero General Public License v3.0

No license ▾

choosealicense.com

- Has more licenses than the ones available by default on GitHub
- Use this option to add it to your GitHub repo

[Home](#) / [Licenses](#)

Creative Commons Attribution Share Alike 4.0 International

[Copy license text to clipboard](#)

Similar to [CC-BY-4.0](#) but requires derivatives be distributed under the same or a similar, [compatible](#) license. Frequently used for media assets and educational materials. A previous version is the default license for Wikipedia and other Wikimedia projects. Not recommended for software.

Permissions

- Commercial use
- Distribution
- Modification
- Private use

Conditions

- License and copyright notice
- Same license
- State changes

Limitations

- Liability
- Patent use
- Trademark use
- Warranty

Suggest this license

Make a pull request to suggest this license for a project that is [not licensed](#). Please be polite: see if a license has already been suggested, try to suggest a license fitting for the project's [community](#), and keep your communication with project maintainers friendly.

Enter GitHub repository URL

How to apply this license

Create a text file (typically named LICENSE or LICENSE.txt) in the root of your source code and copy the text of the license into the file. It is also acceptable to solely supply a link to a copy of the license, usually to the [canonical URL for the license](#).

Optional steps

Add [CC-BY-SA-4.0](#) to your project's package description, if applicable (e.g., [Node.js](#), [Ruby](#), and [Rust](#)). This will ensure the license is displayed in package directories.

[Source](#)

Who's using this license?

[Flight rules for Git](#)
[Material Design Iconic Font](#)
[OWASP MSTG](#)

Attribution-ShareAlike 4.0 International

Creative Commons Corporation ("Creative Commons") is not a law firm and does not provide legal services or legal advice. Distribution of Creative Commons public licenses does not create a lawyer-client or other relationship. Creative Commons makes its licenses and related information available on an "as-is" basis. Creative Commons gives no warranties regarding its licenses, any material licensed under their terms and conditions, or any related information. Creative Commons disclaims all liability for damages resulting from their use to the fullest extent possible.

Using Creative Commons Public Licenses

Creative Commons public licenses provide a standard set of terms and conditions that creators and other rights holders may use to share original works of authorship and other material subject to copyright and certain other rights specified in the public license below. The following considerations are for informational purposes only, are not exhaustive, and do not form part of our licenses.

Considerations for licensors: Our public licenses are intended for use by those authorized to give the public permission to use material in ways otherwise restricted by copyright.

 **RDM-Escape-Room** Public

 Pin

 Watch 0

 Fork 0

 Star 0

 main

 1 Branch

 0 Tags

Go to file

Add file

<> Code

 midnritelibrary	Create .gitignore	2338fc1 · now	2 Commits
 .gitignore	Create .gitignore	now	
 LICENSE	Initial commit	1 minute ago	
 README.md	Initial commit	1 minute ago	

 **README**

 CC0-1.0 license



RDM-Escape-Room

About

No description, website, or topics provided.

-  Readme
-  CC0-1.0 license

-  Activity
-  0 stars
-  0 watching
-  0 forks

Releases

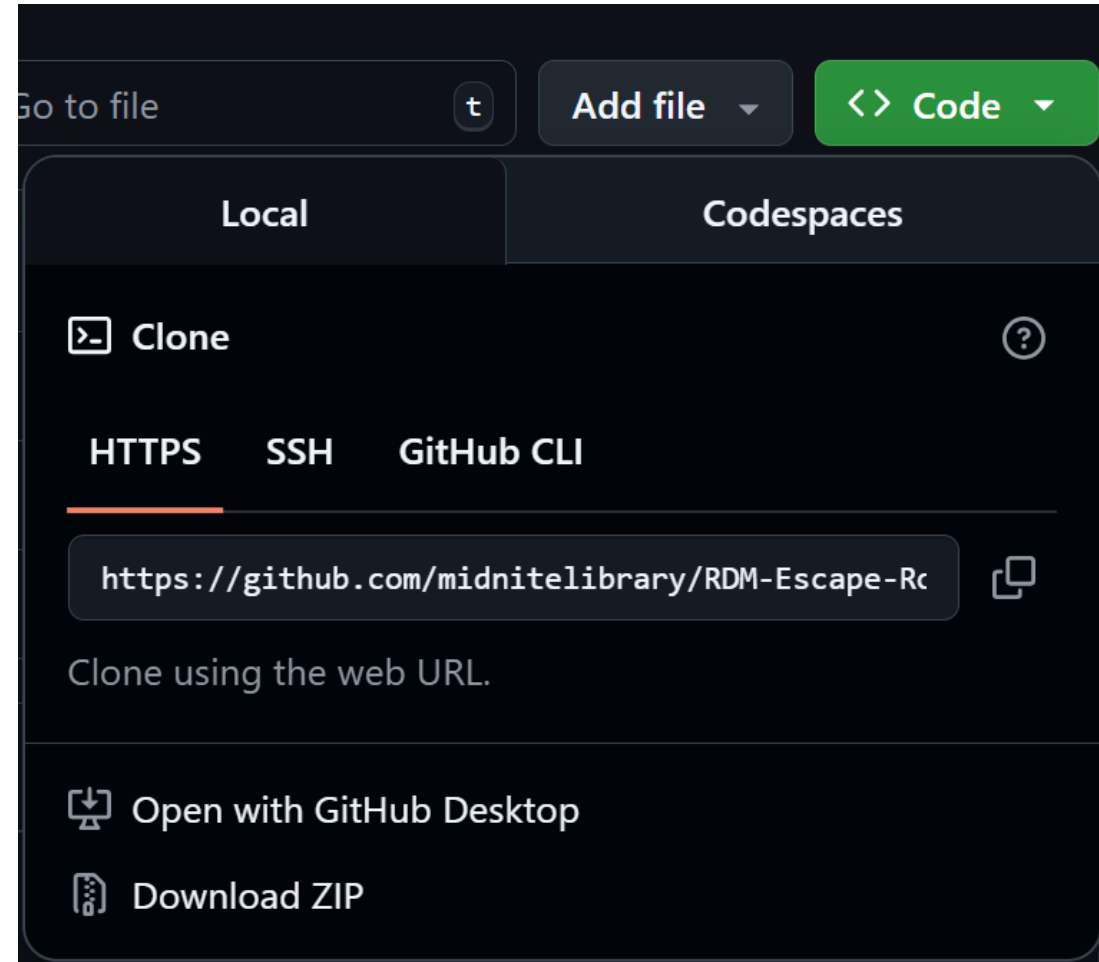
No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

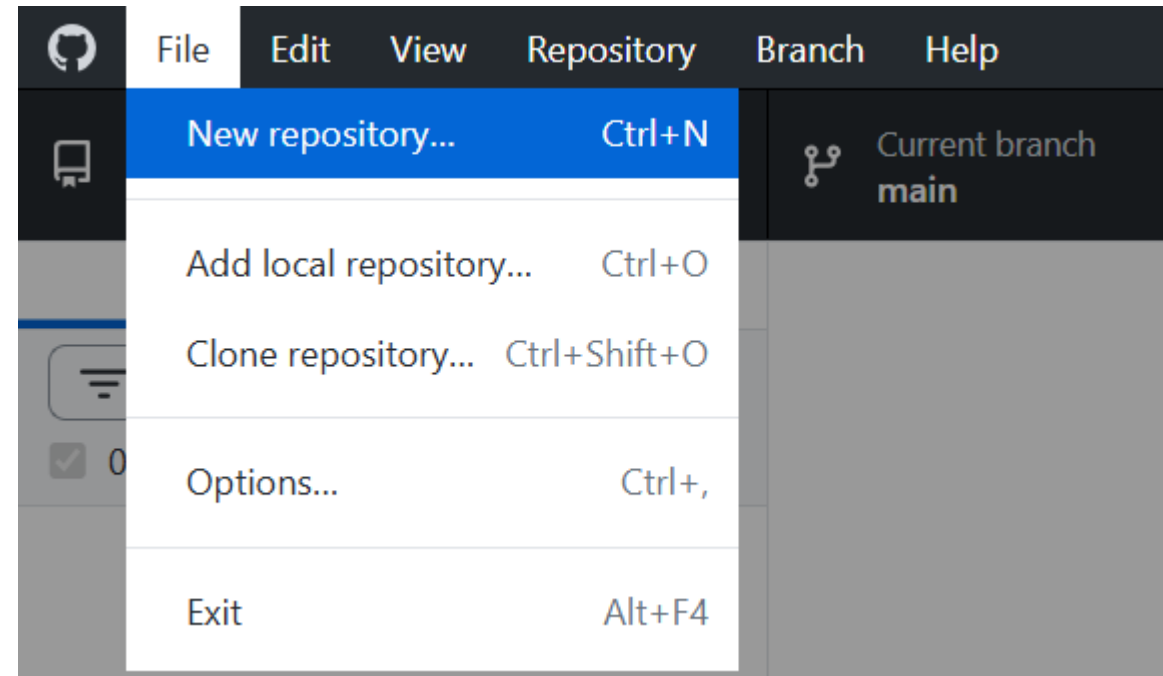
Clone a Repo to Your Computer

- On a GitHub repo click on “Code”
- Copy the URL (or use “Open with GitHub Desktop”)



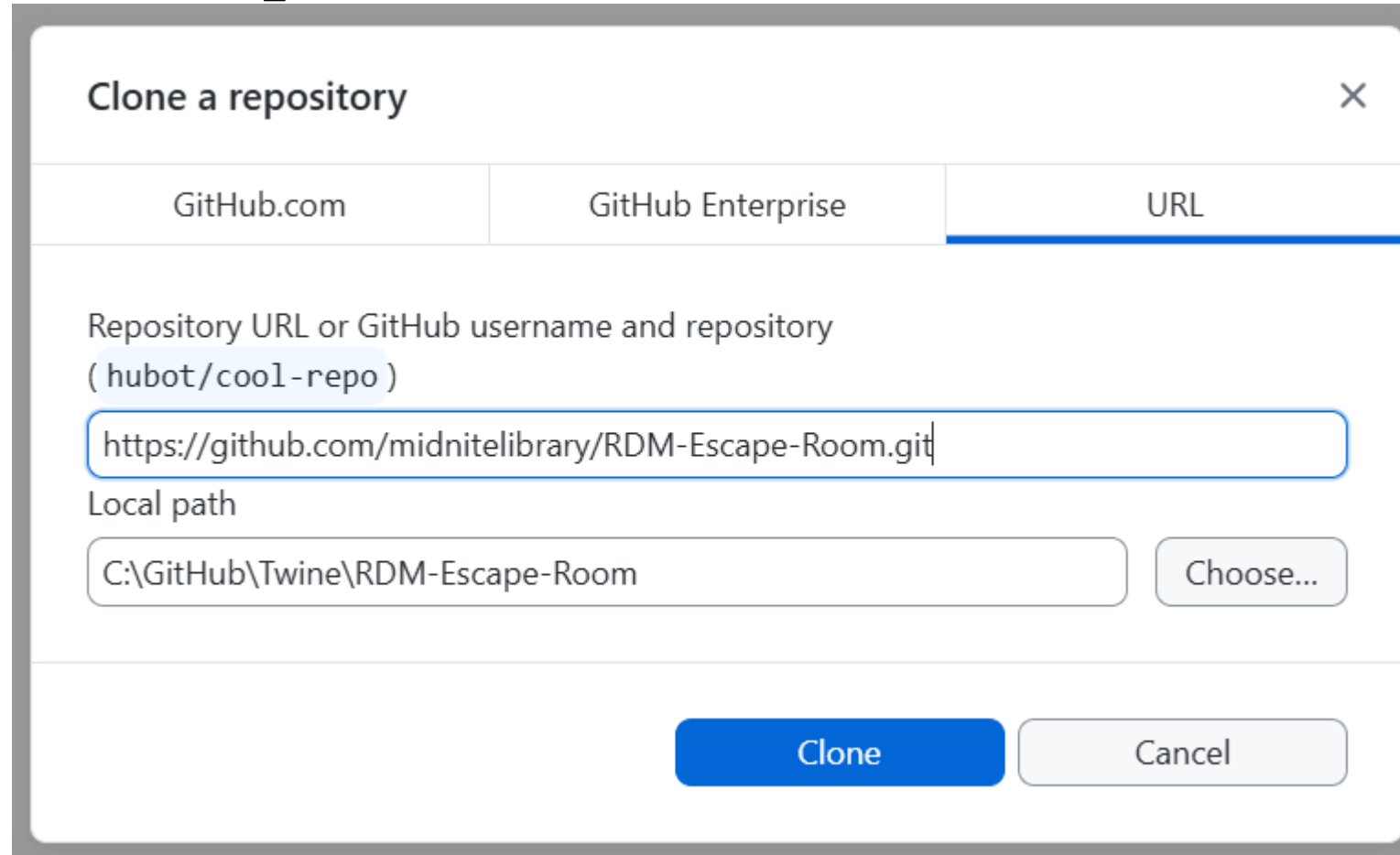
In GitHub Desktop

- Click “File”
- Select “Clone repository”



In GitHub Desktop

- Select “URL”
- Paste in the URL
- Click “Clone”



The screenshot shows the 'Clone a repository' dialog box in GitHub Desktop. It has a title bar with a close button (X). Below the title bar are three tabs: 'GitHub.com', 'GitHub Enterprise', and 'URL'. The 'URL' tab is selected and highlighted with a blue underline. Below the tabs, there is a text input field labeled 'Repository URL or GitHub username and repository (hubot/cool-repo)' containing the URL 'https://github.com/midnritelibrary/RDM-Escape-Room.git'. Below this is a 'Local path' section with a text input field containing 'C:\GitHub\Twine\RDM-Escape-Room' and a 'Choose...' button. At the bottom right are two buttons: 'Clone' (blue) and 'Cancel' (light gray).

Clone a repository

GitHub.com GitHub Enterprise **URL**

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

https://github.com/midnritelibrary/RDM-Escape-Room.git

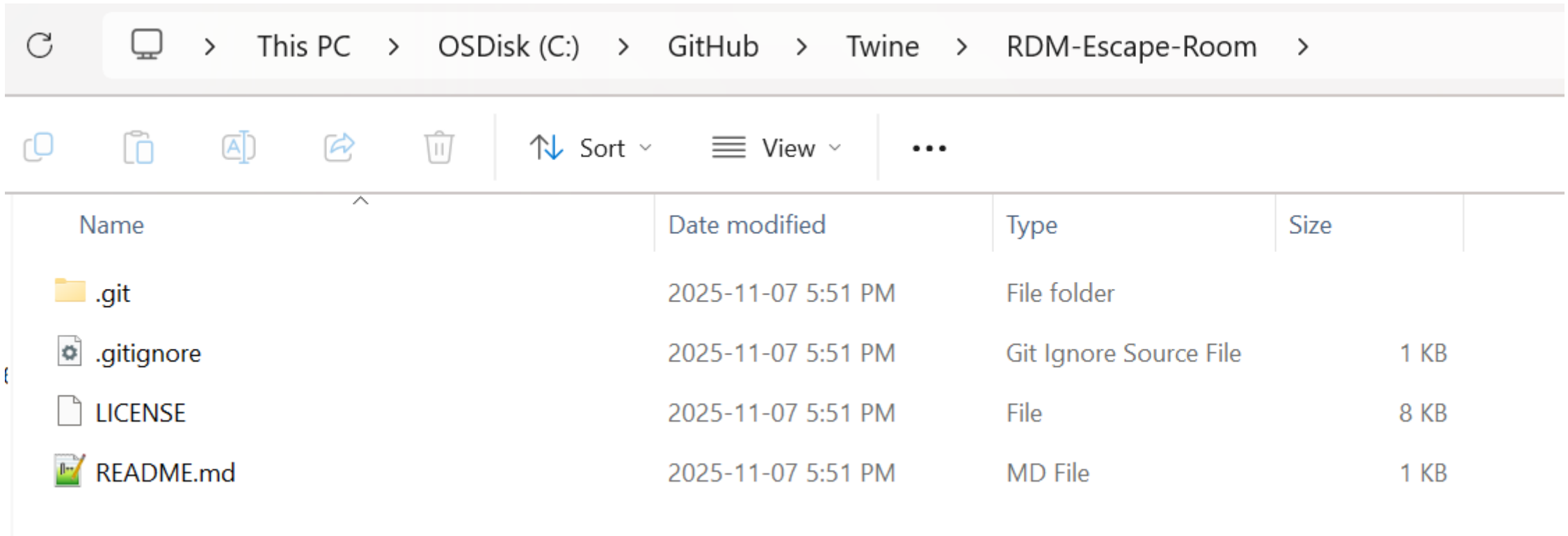
Local path

C:\GitHub\Twine\RDM-Escape-Room Choose...

Clone Cancel

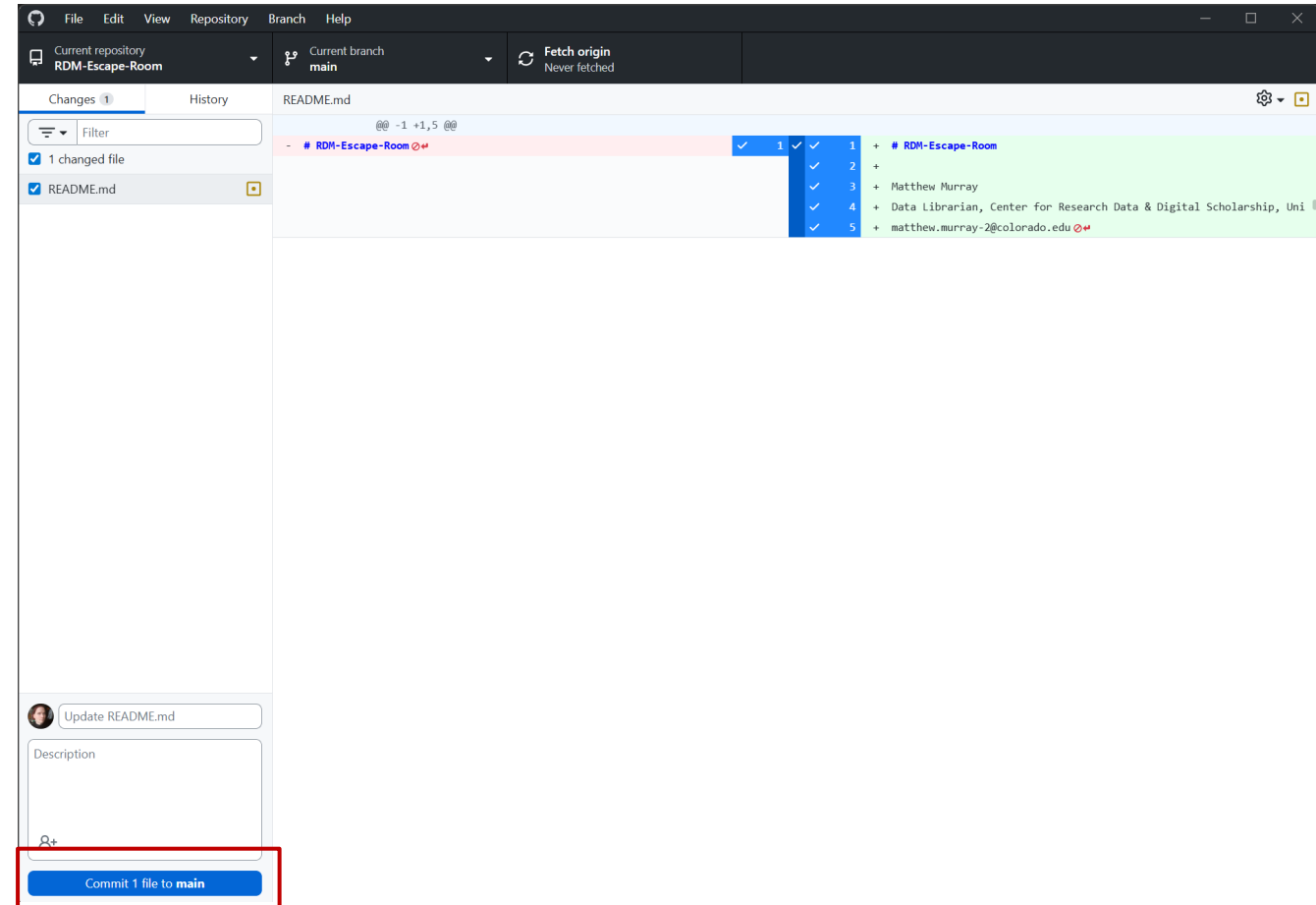
What does this do?

- Copies all files from the GitHub repo to your computer



Updating GitHub Repos

- When you make a change to a file those changes will appear in GitHub
- Click “Commit 1 file to main”
 - Write a description of what changes you made



Updating GitHub Repos

- The commit has been made in Git, but hasn't been pushed to GitHub
- Click “Push origin”

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.



Push commits to the origin remote

You have 1 local commit waiting to be pushed to GitHub.

Always available in the toolbar when there are local commits waiting to be pushed or

Ctrl + P

Push origin

Open the repository in your external editor

Select your editor in [Options](#)

Repository menu or Ctrl + Shift + A

Open in Notepad++

View the files of your repository in Explorer

Repository menu or Ctrl + Shift + F

Show in Explorer

Open the repository page on GitHub in your browser

Repository menu or Ctrl + Shift + G

View on GitHub

main1 Branch0 TagsGo to fileAdd fileCode

midnitelibrary Update README.md

419fa29 · now

4 Commits



.gitignore

Create .gitignore

8 minutes ago



LICENSE

Initial commit

9 minutes ago



README.md

Update README.md

now



README



CC0-1.0 license



RDM-Escape-Room

Matthew Murray

Data Librarian, Center for Research Data & Digital Scholarship, University of Colorado Boulder Libraries

matthew.murray-2@colorado.edu

Help!

Matthew Murray

- Center for Research Data & Digital Scholarship (University Libraries)
- Website: colorado.edu/crdds/
- Helpdesk: crdds@colorado.edu
- Events: colorado.edu/crdds/events
- Interdisciplinary Data Consulting Hours
 - Tuesdays (in person & online)
 - Thursdays (online)

Help! I'm stuck, where do I go?

- Trainings with Center for Research Data and Digital Scholarship (CRDDS): <https://www.colorado.edu/crdds/>
- Software Carpentries tutorial:
<https://swcarpentry.github.io/git-novice/index.html>
- GitHub Student Developer Pack:
<https://education.github.com/pack>
- Helpdesk: rc-help@colorado.edu

Command line walkthrough

Hands on tutorial

Goal: Create a simple project that contains a markdown file

First let's create a new directory for our project:

```
$ mkdir git_work  
$ cd git_work  
$ mkdir git-tutorial  
$ cd git-tutorial
```

Git Repository (Repo)

A Git repository tracks and saves the history of all changes made.

- All of this information is stored in “.git”, which is the repository folder

We can make a directory (folder) a Git repo using “git init”

Git Init

In your “git-tutorial” directory run

```
$ git init
```

- Git creates the "hidden" directory called “.git”

```
$ ls -a
```

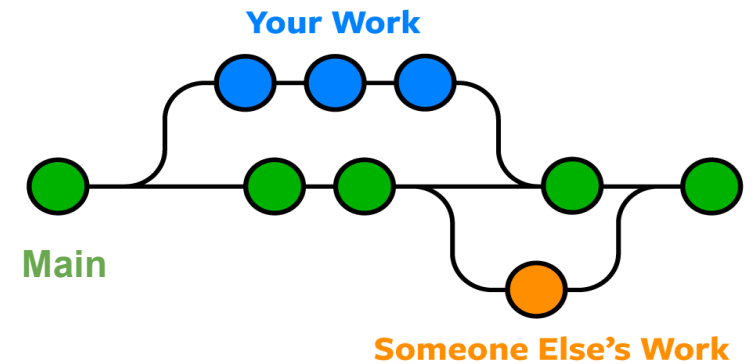
- Your directory is now a repo!
 - Git is now ready to be used
 - Allows us to tell Git what items to watch

Create the main branch

Now that we have a repo, we can create branches. Branches are a version of the repository.

- It is customary to name the primary branch “main”
- This can be done as follows (after an init)
 - `$ git checkout -b main`
- You can switch between branches
 - `$ git checkout <branch-name>`
- To list branches and see what branch you are on use

`$ git branch`



Let's add a file!

It is customary to add a README.md

- Description of repo and any helpful information

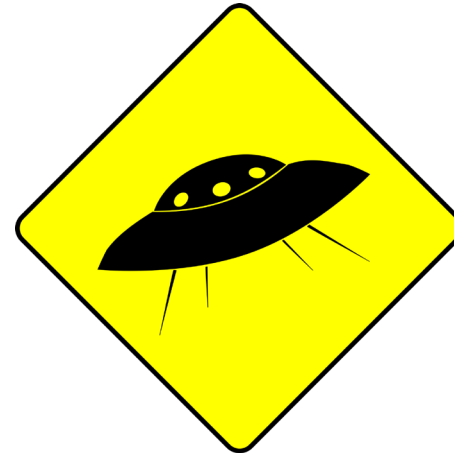
To add a README.md, in “git-tutorial” create and edit the file using nano (or an editor of your choice)

```
$ nano README.md
```

- Add anything you would like!
- Be sure to save the file when you exit.



Git does not know about README.md yet!!



Areas of Git Workflow

Working Area

- Items that you are currently working on
- Are not tracked by Git!
- Exists locally

Staging Area

- When Git starts tracking and saving your work
- Exists locally
- Items are added to this area by using “git add”

Snapshot Area

- All staged items are captured
- Version of the repo
- Exists locally
- Items are added to this area by using “git commit”

GitHub

- Exists locally and on GitHub!
- Items are added to this area using “git push”

Git Status

The git status command **displays the state of the working and staging area.**

Let's see what area README.md is in

```
$ git status
```

- We see it is an untracked file, so it is in the working area

**What if you don't want Git
to track something?**

.gitignore

We can add a file named “.gitignore” to our repo

- Specifies what items (files, directories, etc.) should never be tracked

Let's create a file to ignore!

```
$ echo "Super secret stuff" > confidential_data.txt
```

Add “.gitignore” to “git-tutorial” and put “confidential_data.txt” in it

```
$ echo confidential_data.txt > .gitignore
```

**Let's add our files to the
staging area now!**

Git Add

The git add command **adds a change in the working area to the staging area**

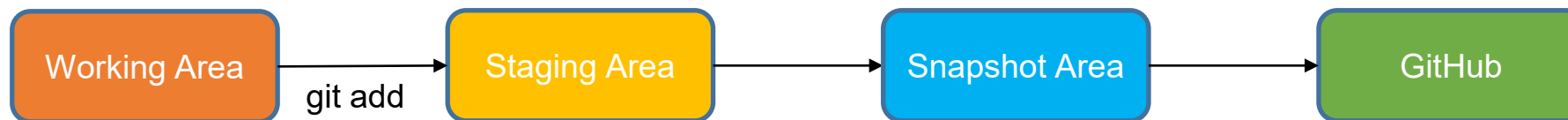
Let's add our README.md to the staging area

```
$ git add README.md
```

or add everything in the current directory

```
$ git add .
```

- Anytime a change is made, you need to do a git add (to track them)



Git Commit

The git commit command **captures a snapshot of all staged items**

- Commits can be thought of as a version of the repo
- Commits should be accompanied with a brief message

Let's commit our staged item!

```
$ git commit -m 'Create repo, add README.md, add .gitignore'
```

```
$ git status
```



Common practice – add, commit

- `git add`
 - Can be performed as much as you want
 - Doesn't need to be done after every change
- `git commit`
 - Always include a comment!!
 - Bundle common staged items together
 - Try not to put too many things in a commit

Git Log

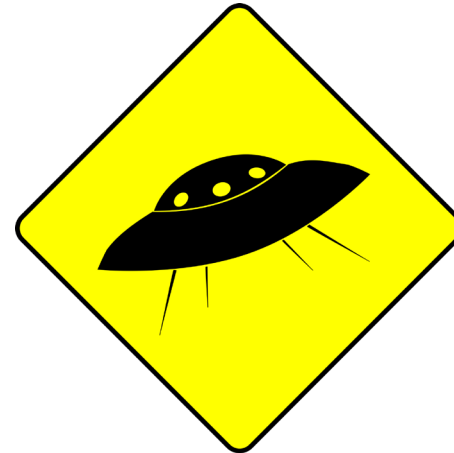
The command git log **lists the commits made in that repository**

- Lists the most recent commits first

\$ git log



All changes and files are only locally stored right now!



To GitHub we go!

GitHub

When you first create a repo locally, you will need to setup a new repository on GitHub too

- Go to <https://github.com>
- Sign in
- Click on “Create New Repository” or just “New”

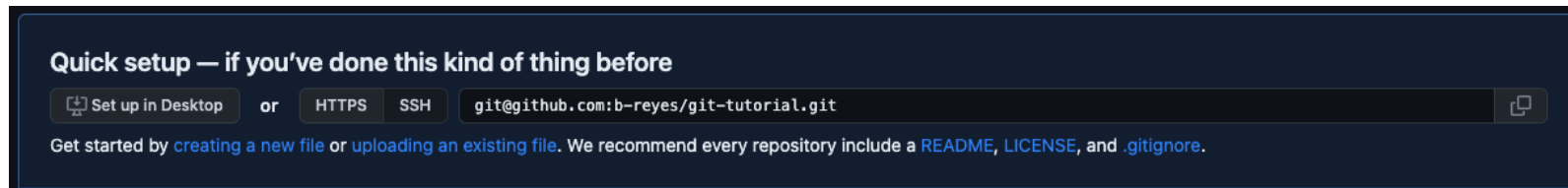
Recent Repositories

 New

Find a repository...

Create Repo in GitHub

- Name your repo, I chose “git-tutorial”
- Don’t add a README or a .gitignore
- Click “Create repository”
- We have set everything up in the previous slides, we only need to copy the ssh link!



Linking local repo to GitHub repo

Git Remote

- Used to identify the remote (e.g. GitHub) repos are linked to your local repo
- Used to link remote repos to your local repo

To view currently linked remote repos:

```
$ git remote -v
```

To link our remote repository:

- When using an SSH key do:

```
$ git remote add origin git@github.com:<user>/git-tutorial.git
```

- When using a Personal Access Token do:

```
$ git remote add origin https://github.com/<user>/git-tutorial.git
```

Sending local changes to GitHub

Git Push

Uploads local repository content to a remote repository

- Pushing is how you transfer commits from your local repo to a remote repo

```
$ git push <name of remote repo> <branch>
```

```
$ git push origin main
```



GitHub

- Go back to GitHub and refresh your page
 - should see the files we have added (and not the ones we've ignored)
- Some cool features!
 - look at our commits
 - directly edit/commit in the browser
- Let's do that! Let's something and commit it on GitHub
 - But now our remote repo is one commit ahead of our local one...

Git Fetch & Merge

- Git fetch retrieves the changes from the remote repo

```
$ git fetch
```

- Git merge combines two branches

```
$ git merge origin/main
```

There's an easier way!

Git Pull

Git pull combines the fetch and merge commands

```
$ git pull <name of remote repo> <branch>
```

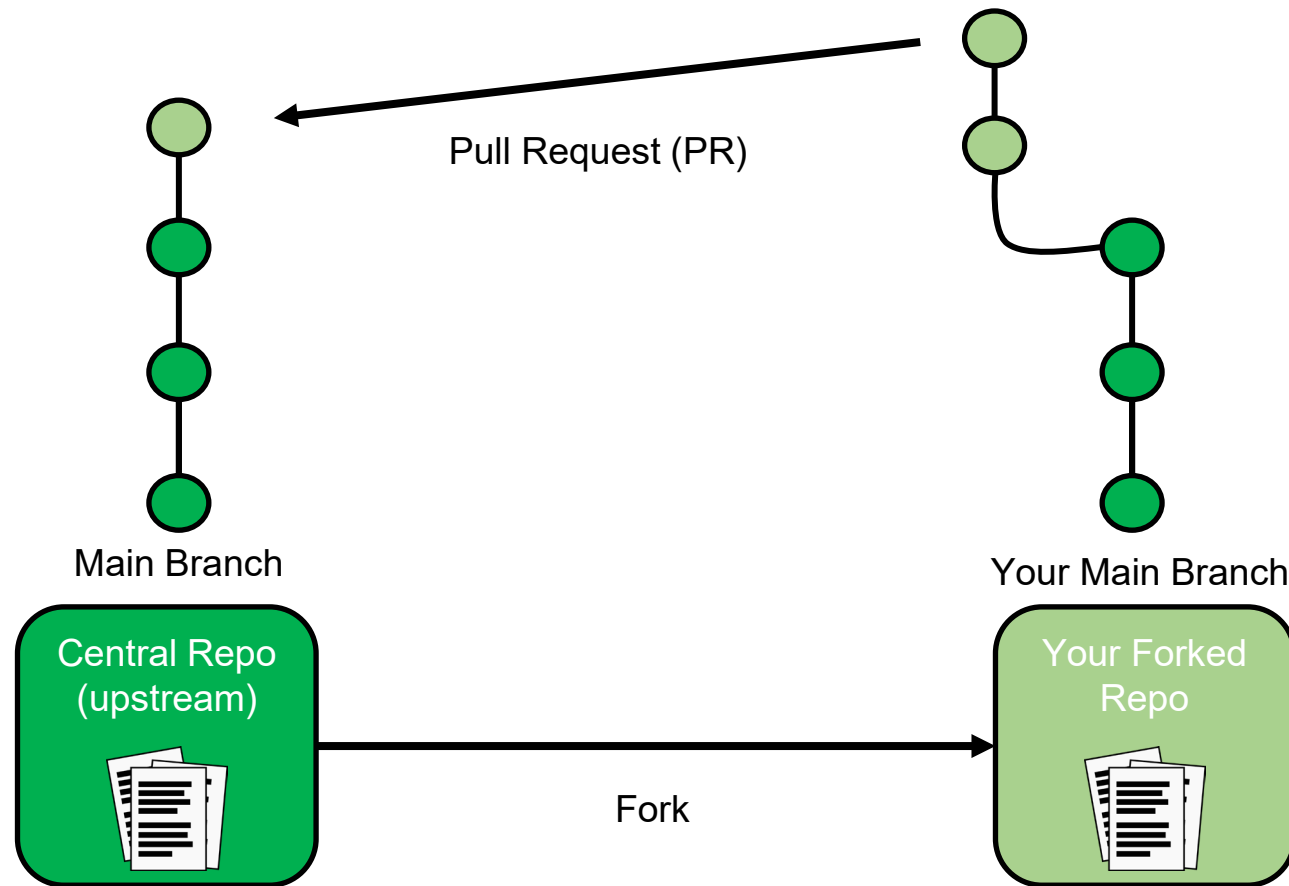
```
$ git pull origin main
```

IMPORTANT!

- Make sure no commits have been done on local branch
- It is fine to have staged items (git add)
- ALWAYS do git pull before any commits!

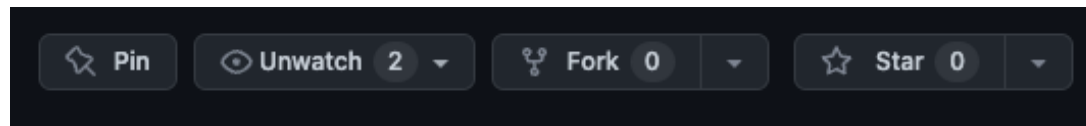
Advanced topic: Collaboration

GitHub Forks



GitHub Forks

- Improves collaboration
 - Don't have to worry about disturbing the upstream repo
 - Improves transparency through pull requests
- Go ahead and Fork my repo:
 - Go to <https://github.com/mohal-k/git-tutorial>
 - Click “Fork” button
 - Click “Create fork”
- Creates your own version of my repo under your GitHub



Git Clone

- Git clone **makes a clone (or copy) of a remote repo in a new directory, at another location.**

```
$ git clone <url> <optional new name>
```

- Easy way to grab third-party code, or pre-existing code you might need to work on
- Cloning when you have SSH keys (be sure to make “git_work_cloned”):

```
$ cd git_work_cloned
```

```
$ git clone git@github.com:<user>/git-tutorial.git
```

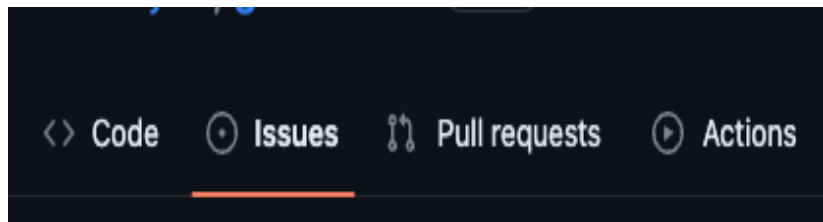
- Cloning when you are using Personal Access Tokens (be sure to make “git_work_cloned”):

```
$ cd git_work_cloned
```

```
$ git clone https://github.com/<user>/git-tutorial.git
```

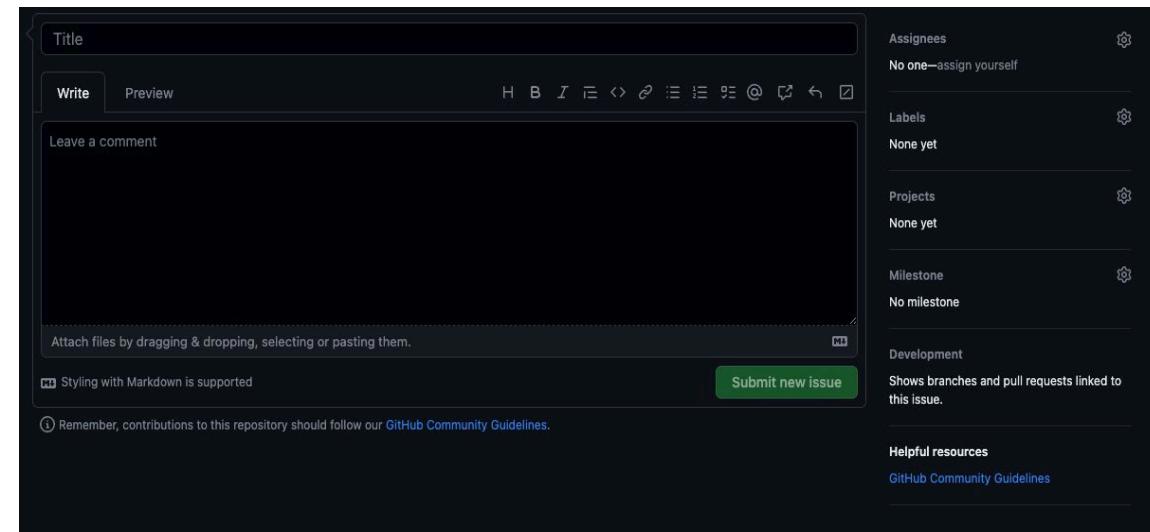
GitHub Issues

- Allows you to discuss the project
- Point out issues, request features, ask for help
- Useful place to see past user discussion



GitHub Issues

- Include as much detail as possible
 - Version of software
 - Operating system
- Provide a simple minimal example, if possible
- If a feature request
 - Outline possible implementation
 - Highlight its value



The image shows the GitHub 'New Issue' form interface. It features a 'Title' input field at the top. Below it are 'Write' and 'Preview' tabs, with a rich text editor toolbar containing icons for bold, italic, code, link, list, and other formatting options. The main body of the form is a large text area with the placeholder text 'Leave a comment'. Below the text area is a file upload section with the text 'Attach files by dragging & dropping, selecting or pasting them.' and a 'Submit new issue' button. At the bottom, there is a note about Markdown support and a link to the GitHub Community Guidelines. On the right side, there are sections for 'Assignees' (showing 'No one—assign yourself'), 'Labels' (showing 'None yet'), 'Projects' (showing 'None yet'), 'Milestone' (showing 'No milestone'), and 'Development' (showing 'Shows branches and pull requests linked to this issue.'). At the bottom right, there is a 'Helpful resources' section with a link to 'GitHub Community Guidelines'.

Pull Requests (PRs)

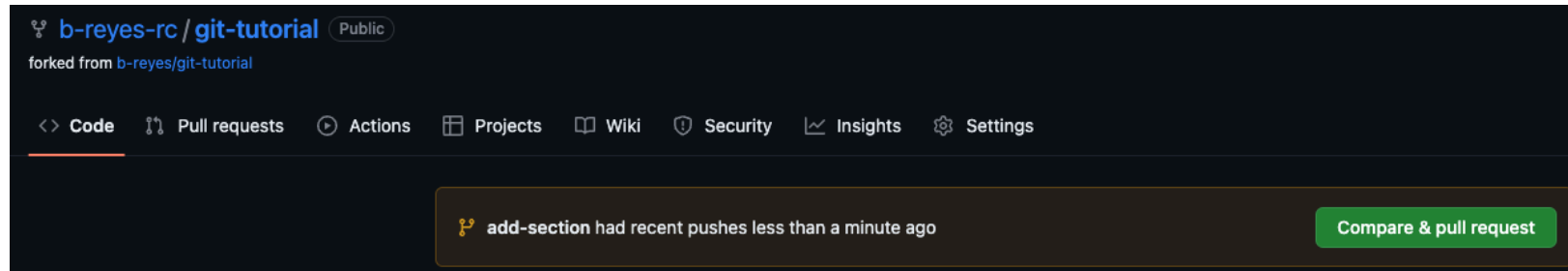
- A request that an upstream repo pull your branch into their branch
- Starting a PR does not automatically merge changes
 - Notifies maintainers of upstream repo
 - Allows maintainers to review your changes
 - Discussion of changes
 - Requested additional changes
- Maintainers of upstream repo merge in the changes

PR steps

1. Fork upstream repo
2. Clone the forked repo
3. Connect forked and cloned repo to upstream repo. Ex. using SSH keys:
`$ git remote add upstream git@github.com:dev-mohalkh/git-tutorial.git`
4. Create a new branch specific to your change
`$ git checkout -b <new-branch> <branch-to-copy>`
5. Make your changes on this branch
6. Perform a git add, commit, and push to origin
7. Create a PR from GitHub

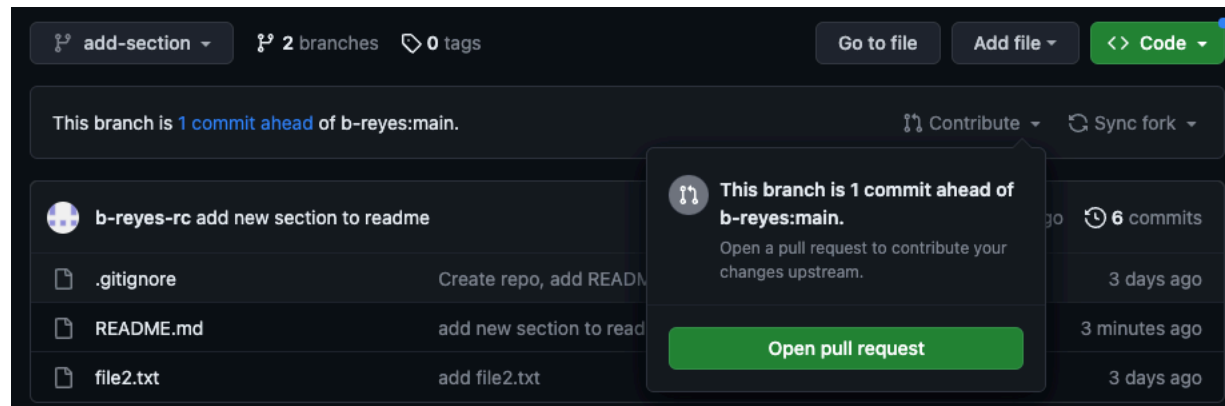
Creating a PR

- After you push your changes to the forked repo, you can click the pop-up “Compare & pull request” on GitHub
 - Will disappear after some time



Creating a PR

- After you push your changes to the forked repo
 1. Switch to your new branch
 2. Click the drop-down arrow next to “Contribute”
 3. Click “Open pull request”
- Will NOT disappear!



Creating a PR

Upstream repo and branch we want to merge into

Our forked repo and branch with our changes

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 repository: **b-reyes/git-tutorial** base: **main** head repository: **b-reyes-rc/git-tutorial** compare: **add-section**
✓ **Able to merge.** These branches can be automatically merged.

 **Add hobby section to README**

Write **Preview** H B I ≡ <> 🔗 ≡ ≡ ≡ @ ↻ ↩

Leave a comment

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

☒ **Allow edits by maintainers** ⓘ

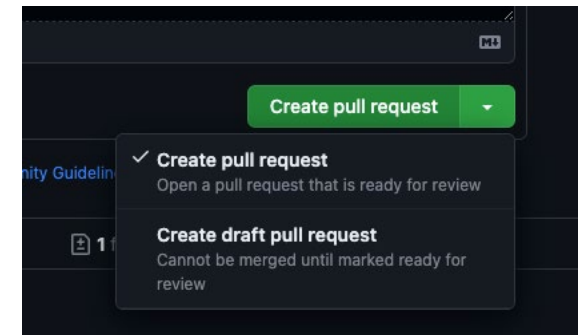
Create pull request ▾

ⓘ Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Helpful resources
[GitHub Community Guidelines](#)

Pull Requests – Best Practice

- Create a new feature branch of forked repo
- When submitting a PR
 - Provide a short descriptive title
 - In comment section
 - Link to any current issue
 - Describe what the PR does and reasons for it
- Draft pull requests
 - PR is a work in progress
 - Can be used for discussion



Merging

- When doing a “git pull” you are merging in changes
- This process can be done manually
- When collaborating, multiple individuals can be working on the same item
 - Conflicts can happen!
- One needs to manually resolve conflicts
- Fantastic tutorial on merging:

<https://www.atlassian.com/git/tutorials/using-branches/git-merge>