git and github

Web Development 1
john.rellis@setu.ie

git guide – useful for self learning

https://github.com/git-guides

Install git

We are going to use the command line and visual studio code – this means, if you are on windows, you will use a program called git bash.

On other systems, you can use the native terminal.

If you choose to use github-cli, you are on your own.

Do not install a GUI program yet.

Install git

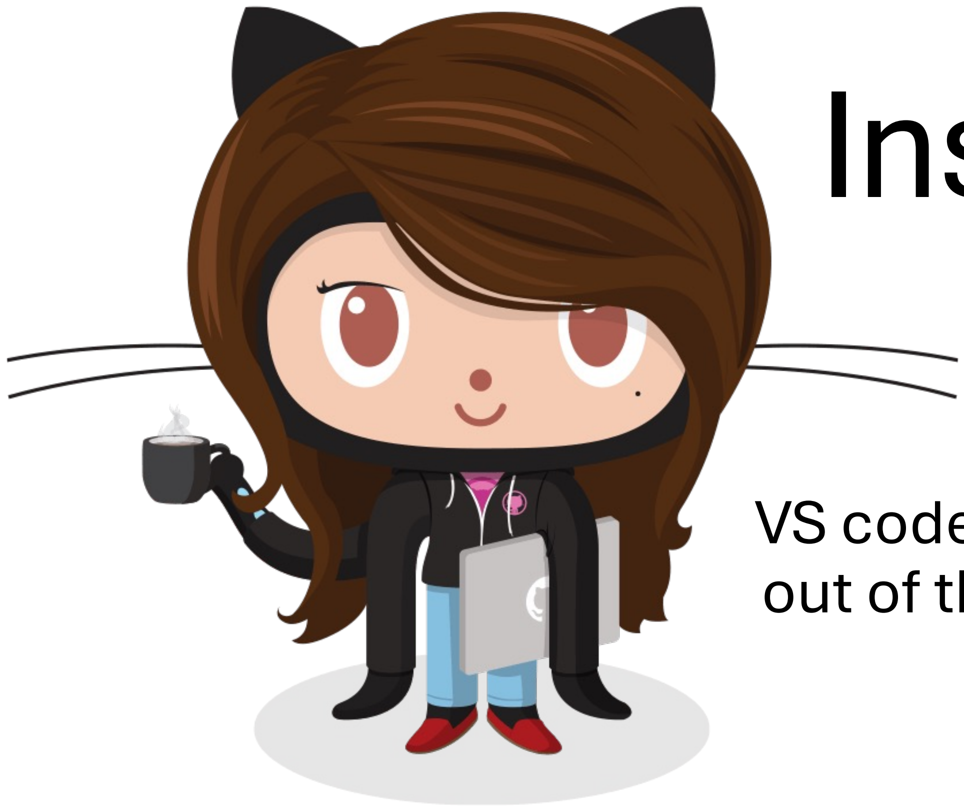
https://github.com/git-guides/install-git

Configure your laptop

- Set your git username as per:
  - https://docs.github.com/en/get-started/getting-started-with-git/setting-your-username-in-git
- Set your git email address as per:
  - https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-personal-account-on-github/managing-email-preferences/setting-your-commit-email-address
- If starting out, don't worry too much about config per repository, set everything global, e.g.
  - git config --global user.email "YOUR_EMAIL"
  - git config --global user.name "Mona Lisa"
- Confirm with
  - git config --global user.name
  - git config --global user.email

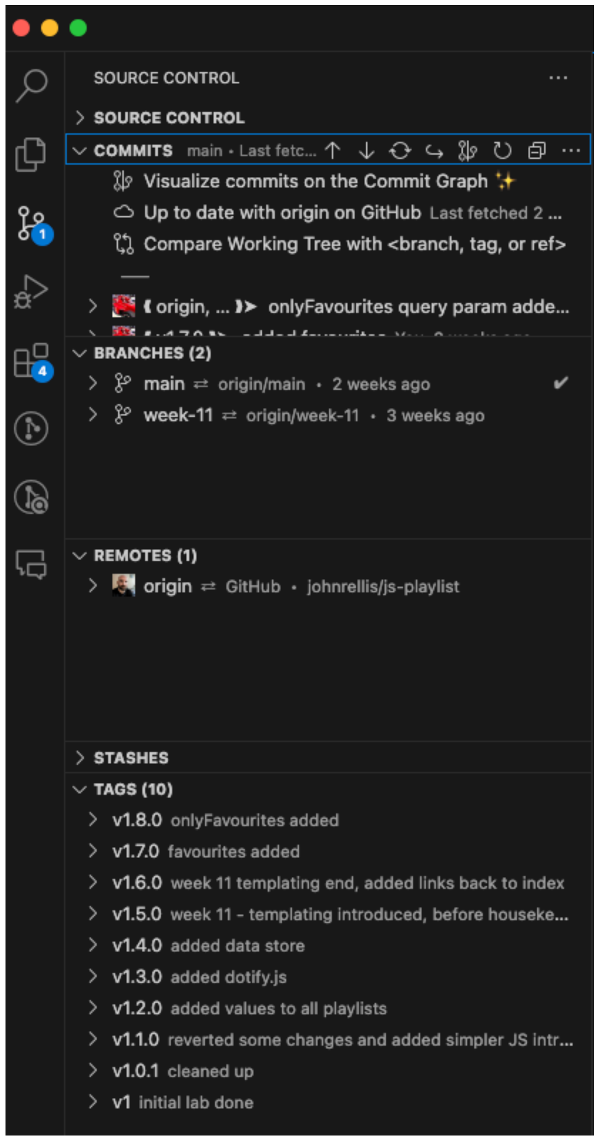
Install vs code

https://code.visualstudio.com/download



Install vs code

VS code supports a host of git features out of the box so we'll work with those first



SOURCE CONTROL ...

> SOURCE CONTROL

COMMITS main • Last fetc...

Visualize commits on the Commit Graph ✨

Up to date with origin on GitHub Last fetched 2 ...

Compare Working Tree with <branch, tag, or ref>

> (origin, ...)> onlyFavourites query param adde...

BRANCHES (2)

> main ⇄ origin/main • 2 weeks ago ✔

> week-11 ⇄ origin/week-11 • 3 weeks ago

REMOTES (1)

> origin ⇄ GitHub • johnrellis/js-playlist

> STASHES

TAGS (10)

> v1.8.0 onlyFavourites added

> v1.7.0 favourites added

> v1.6.0 week 11 templating end, added links back to index

> v1.5.0 week 11 - templating introduced, before houseke...

> v1.4.0 added data store

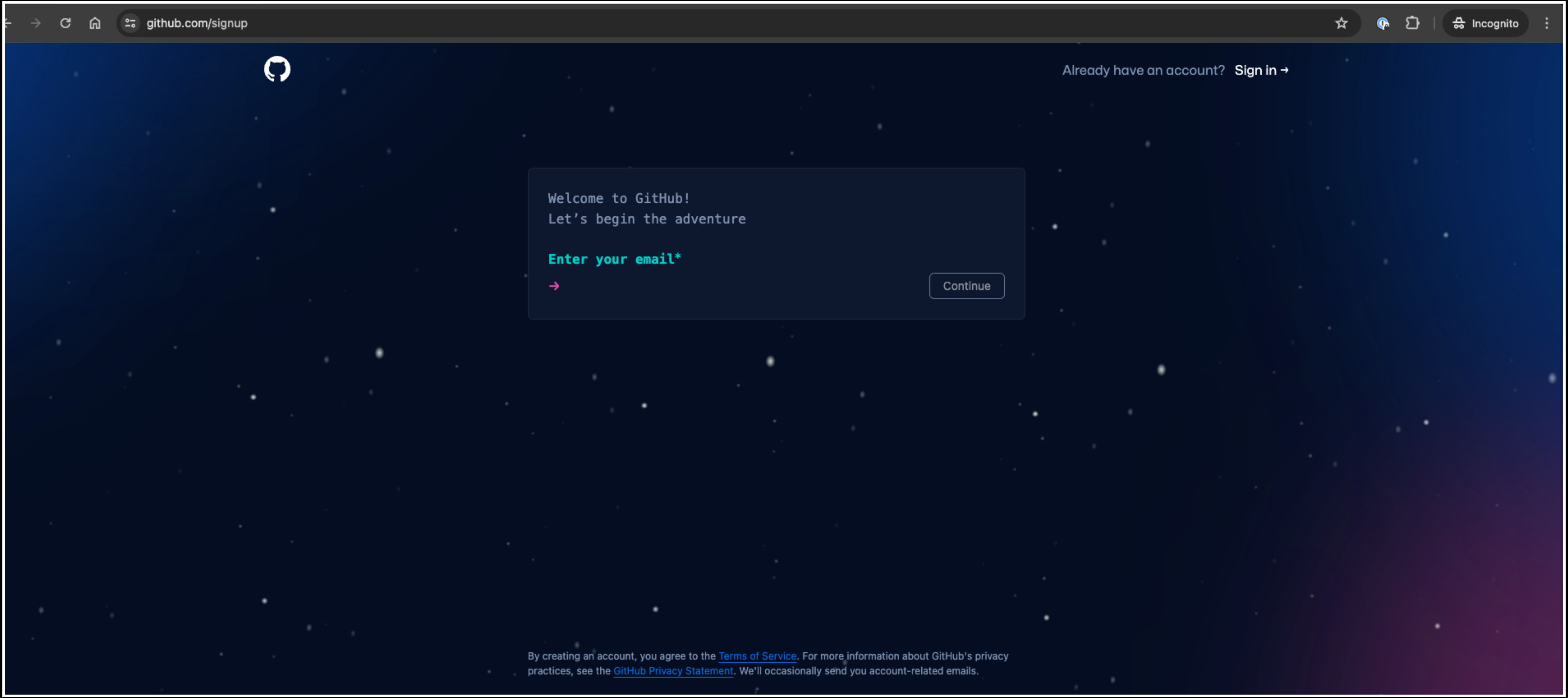
> v1.3.0 added dotify.js

> v1.2.0 added values to all playlists

> v1.1.0 reverted some changes and added simpler JS intr...

> v1.0.1 cleaned up

> v1 initial lab done



Welcome to GitHub!
Let's begin the adventure

Enter your email*

→

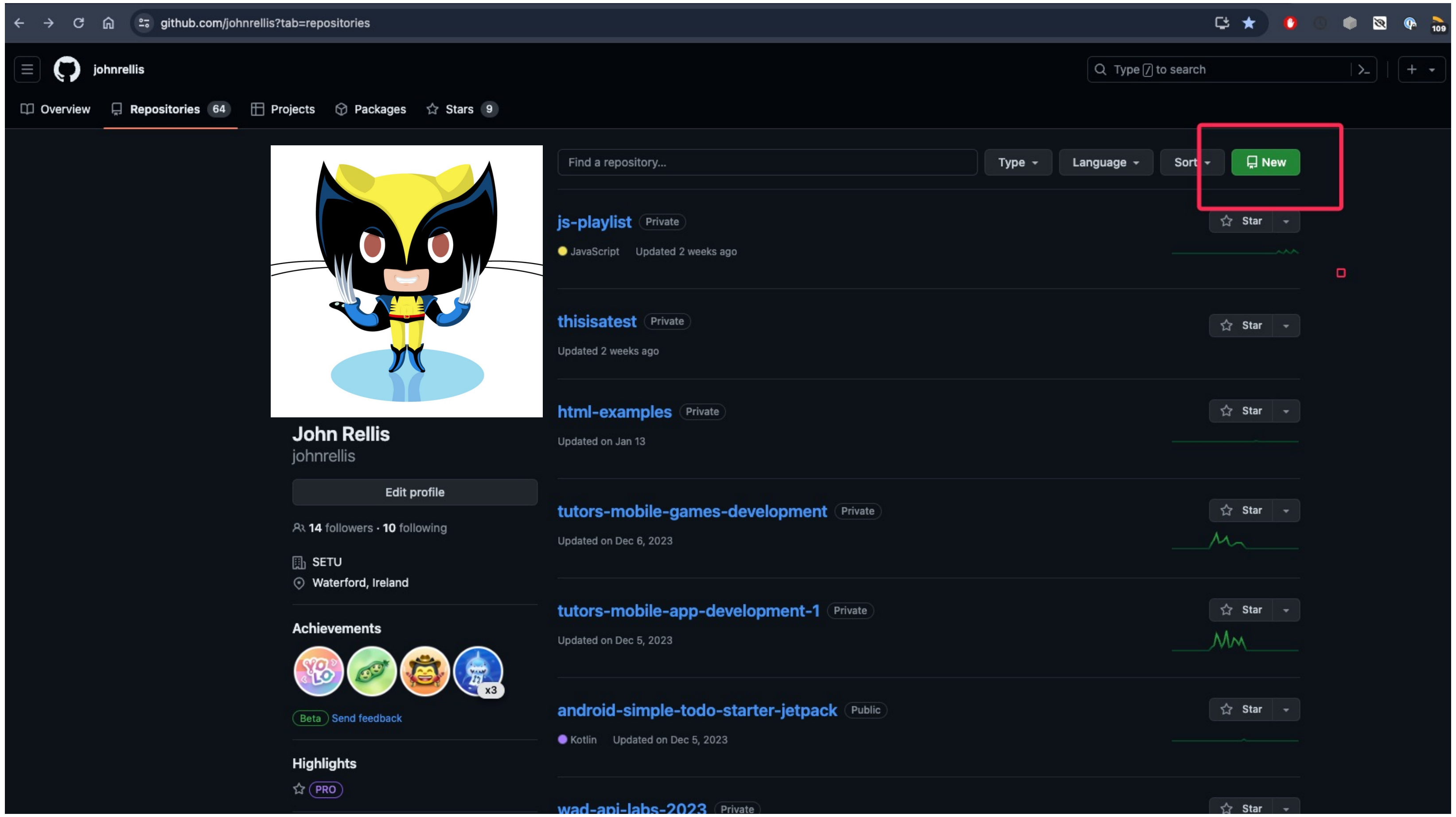
By creating an account, you agree to the Terms of Service. For more information about GitHub's privacy practices, see the GitHub Privacy Statement. We'll occasionally send you account-related emails.

Sign up for a github account – github.com/signup

Public and Private repositories

- You can restrict who has access to a repository by choosing a repository's visibility: public or private.
- When you create a repository, you can choose to make the repository public or private.
- I recommend you start all your repositories private until you have learned enough about software licencing OR you are asked to create public repositories by a teacher/lecturer

Create a Repository / Create a Repo



John Rellis
johnrellis

14 followers · 10 following

SETU
Waterford, Ireland

js-playlist Private
JavaScript Updated 2 weeks ago

thisisatest Private
Updated 2 weeks ago

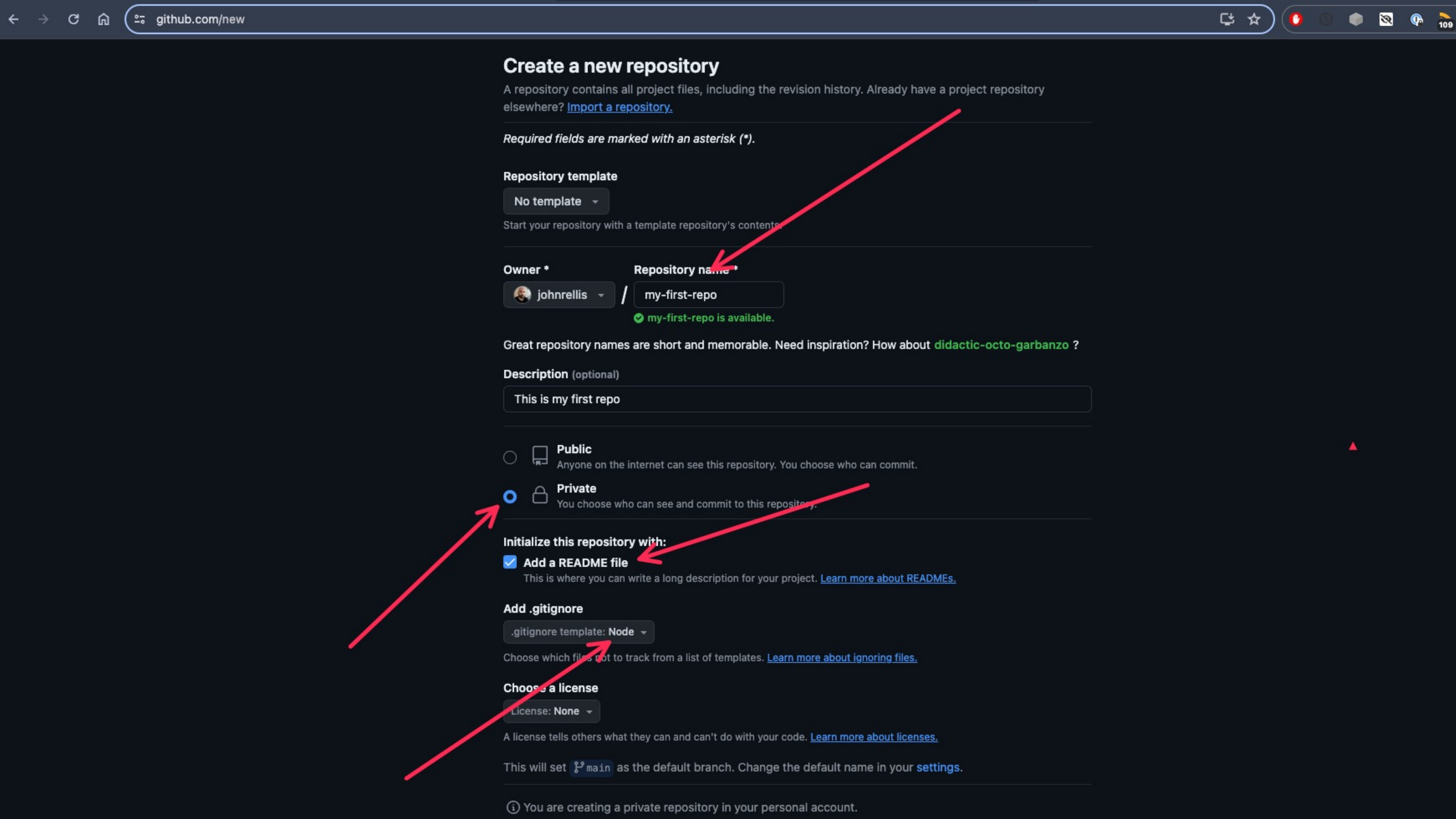
html-examples Private
Updated on Jan 13

tutors-mobile-games-development Private
Updated on Dec 6, 2023

tutors-mobile-app-development-1 Private
Updated on Dec 5, 2023

android-simple-todo-starter-jetpack Public
Kotlin Updated on Dec 5, 2023

wad-api-labs-2023 Private



github.com/new

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Repository template

No template

Start your repository with a template repository's contents.

Owner * johnrellis / **Repository name** * my-first-repo

my-first-repo is available.

Great repository names are short and memorable. Need inspiration? How about didactic-octo-garbanzo ?

Description (optional)

This is my first repo

( ) **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:

[✓] **Add a README file**
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore

.gitignore template: Node

Choose which files not to track from a list of templates. Learn more about ignoring files.

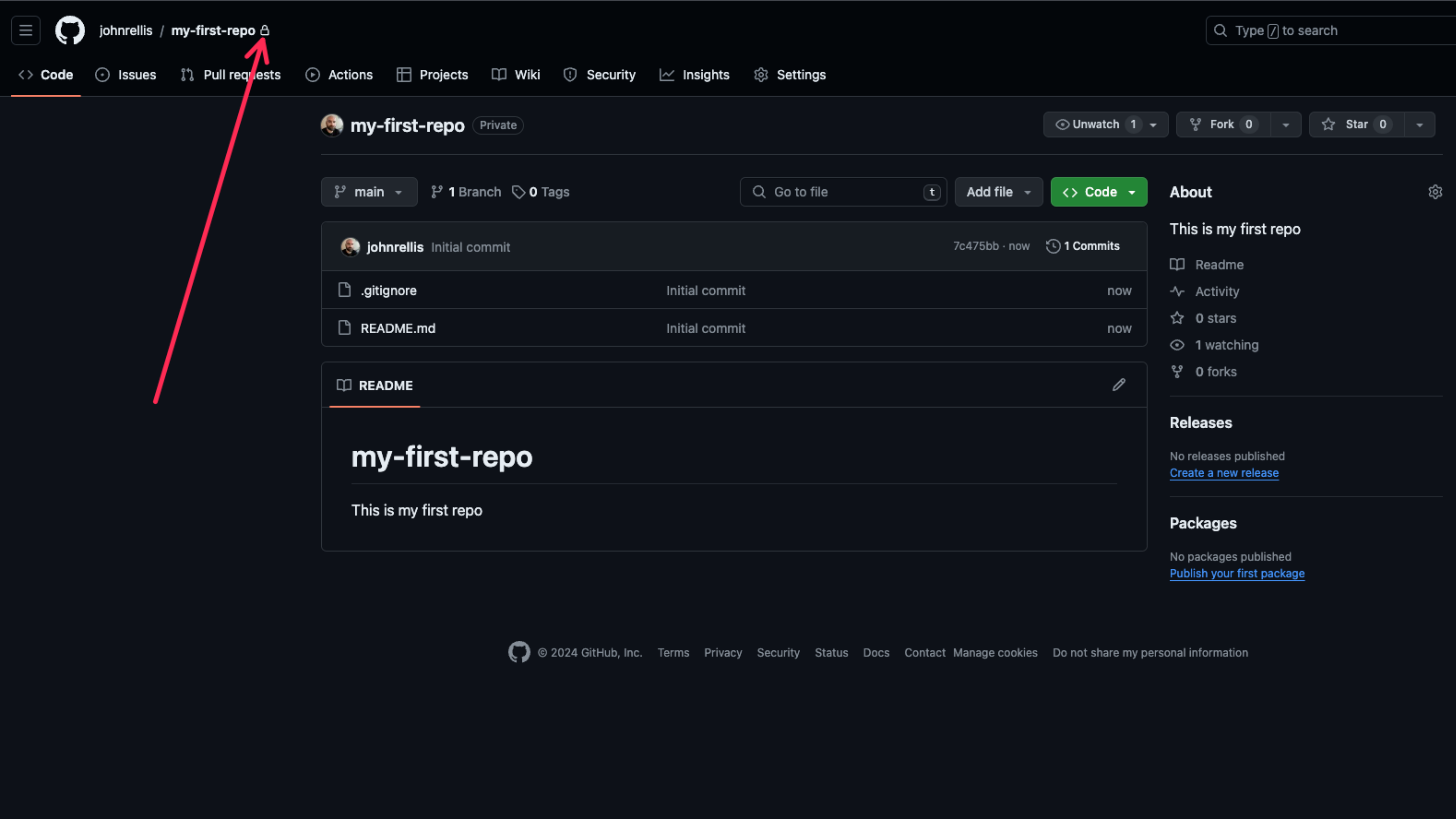
Choose a license

License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

This will set main as the default branch. Change the default name in your settings.

You are creating a private repository in your personal account.



my-first-repo Private

johnrellis Initial commit 7c475bb · now 1 Commits

.gitignore	Initial commit	now
README.md	Initial commit	now

README

my-first-repo

This is my first repo

About

This is my first repo

Readme
Activity
0 stars
1 watching
0 forks

Releases

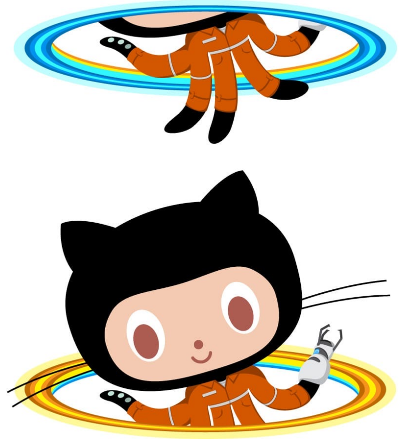
No releases published
Create a new release

Packages

No packages published
Publish your first package

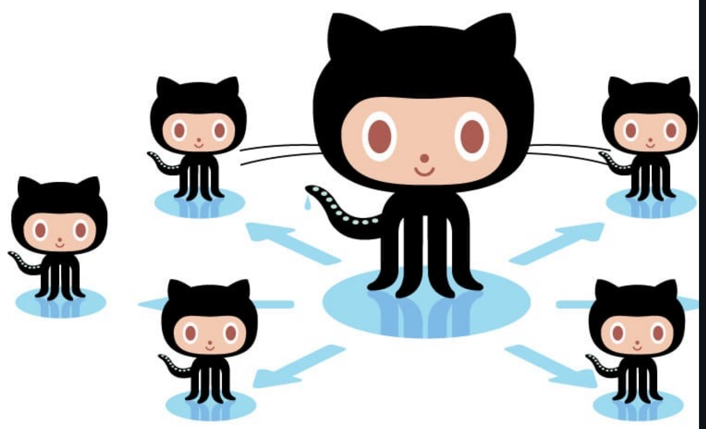
© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

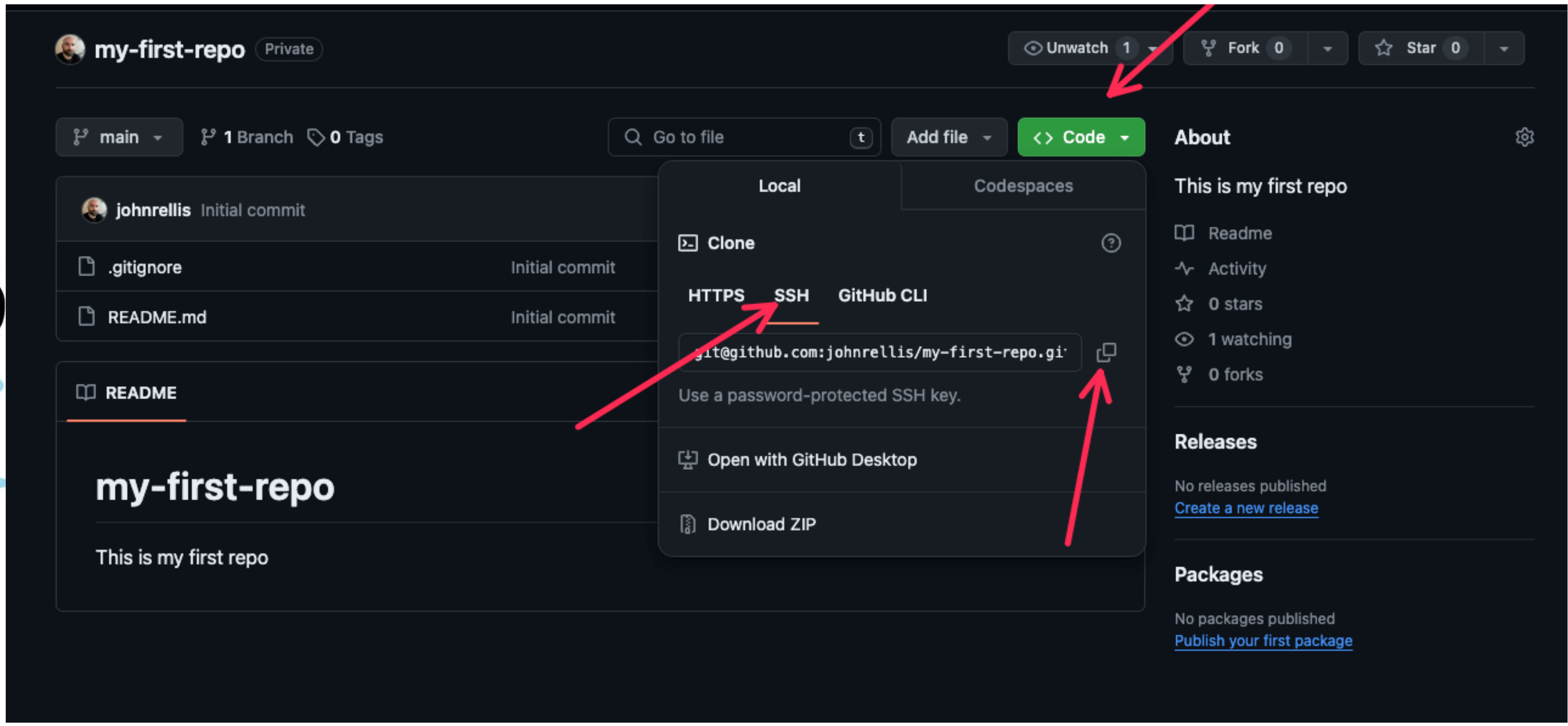
Now for the messy bit….. Authentication



- We are going to use SSH keys to authenticate
  - https://docs.github.com/en/authentication/connecting-to-github-with-ssh/about-ssh
- Read the above and you can jump to the the following steps, note that there are tabs for mac, windows and linux:
  - https://docs.github.com/en/authentication/connecting-to-github-with-ssh/checking-for-existing-ssh-keys
  - https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent
  - https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account
  - https://docs.github.com/en/authentication/connecting-to-github-with-ssh/testing-your-ssh-connection
  - https://docs.github.com/en/authentication/connecting-to-github-with-ssh/working-with-ssh-key-passphrases

If you’ve made it this far, time to clone...



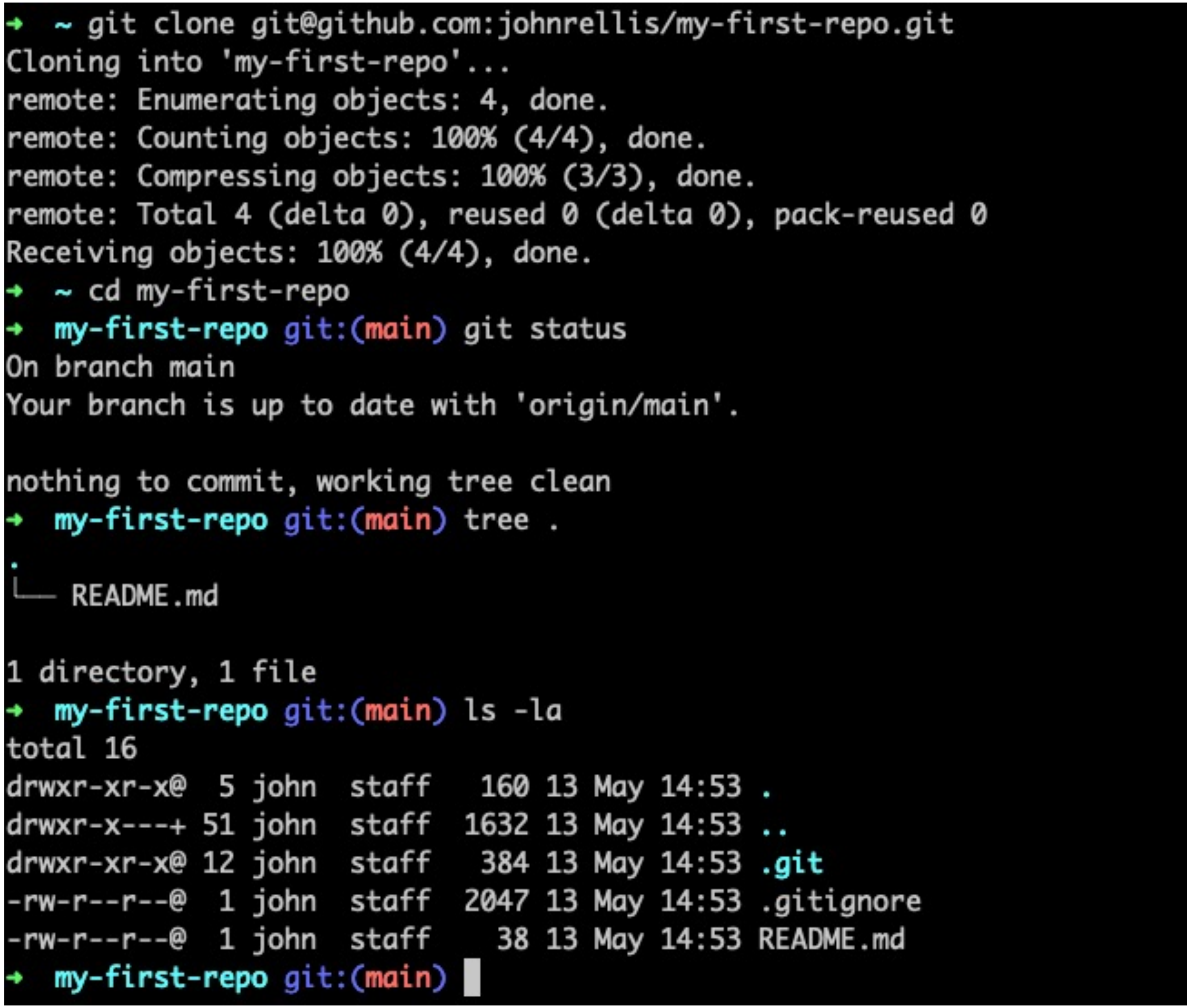


my-first-repo

This is my first repo

```
git@github.com:johnrellis/my-first-repo.gi
```

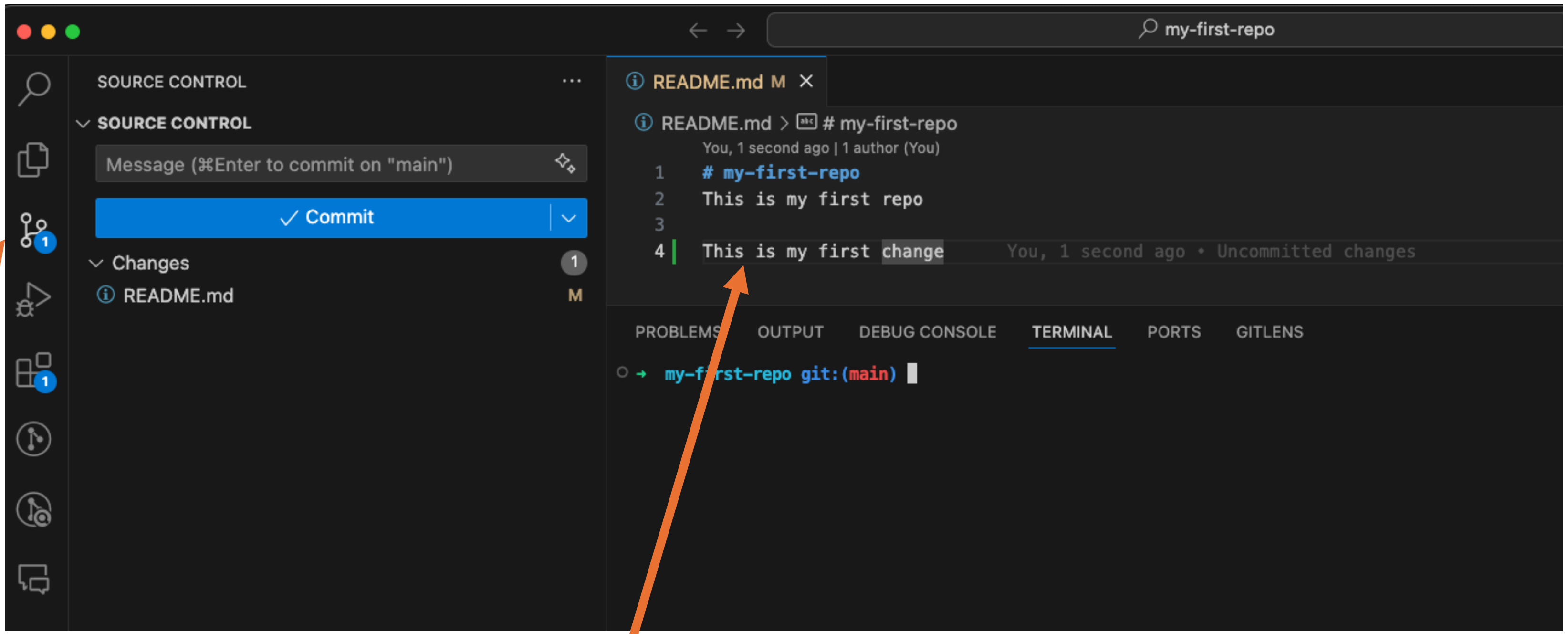
Use a password-protected SSH key.



```
➜  ~ git clone git@github.com:johnrellis/my-first-repo.git
Cloning into 'my-first-repo'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
➜  ~ cd my-first-repo
➜  my-first-repo git:(main) git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
➜  my-first-repo git:(main) tree .
.
└── README.md

1 directory, 1 file
➜  my-first-repo git:(main) ls -la
total 16
drwxr-xr-x@  5 john  staff   160 13 May 14:53 .
drwxr-x---+ 51 john  staff  1632 13 May 14:53 ..
drwxr-xr-x@ 12 john  staff   384 13 May 14:53 .git
-rw-r--r--@  1 john  staff  2047 13 May 14:53 .gitignore
-rw-r--r--@  1 john  staff    38 13 May 14:53 README.md
➜  my-first-repo git:(main) 
```

1. We clone the repo using the copied git URL
2. We cd into my-first-repo
3. git status confirms we have our repo and we are on the main branch (huh, what's a branch?)
4. I have a fancy terminal configuration that shows me git repo information in the command line, you may not.
5. I use tree to inspect the repo, you may not have tree installed, don't worry.
6. ls –la lists all our files, including our hidden files
    1. .git directory, contains all our repo information
    2. .gitignore, our ignore file that was created on repo creation
    3. Our README.md

Open the directory/repo in VS Code



```
# my-first-repo
This is my first repo

This is my first change      You, 1 second ago • Uncommitted changes
```

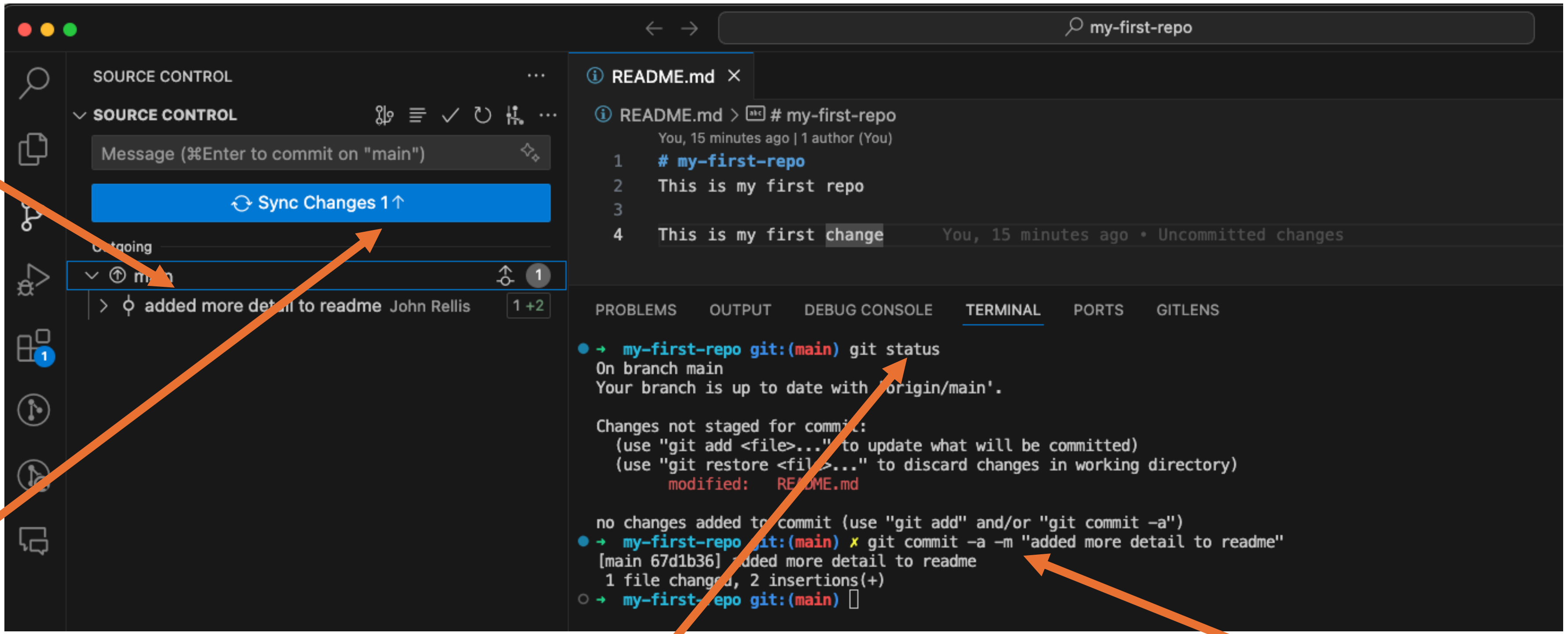
```
○ → my-first-repo git:(main)
```

2. Open the git section

1. Make this change to README.md and save.

Open the directory/repo in VS Code

4. Our changes waiting to be pushed


my-first-repo
SOURCE CONTROL
SOURCE CONTROL
Message (⌘Enter to commit on "main")
Sync Changes 1↑
Outgoing
main
added more detail to readme John Rellis 1 +2
README.md
README.md > # my-first-repo
You, 15 minutes ago | 1 author (You)
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS

```
# my-first-repo
This is my first repo

This is my first change        You, 15 minutes ago • Uncommitted changes
```

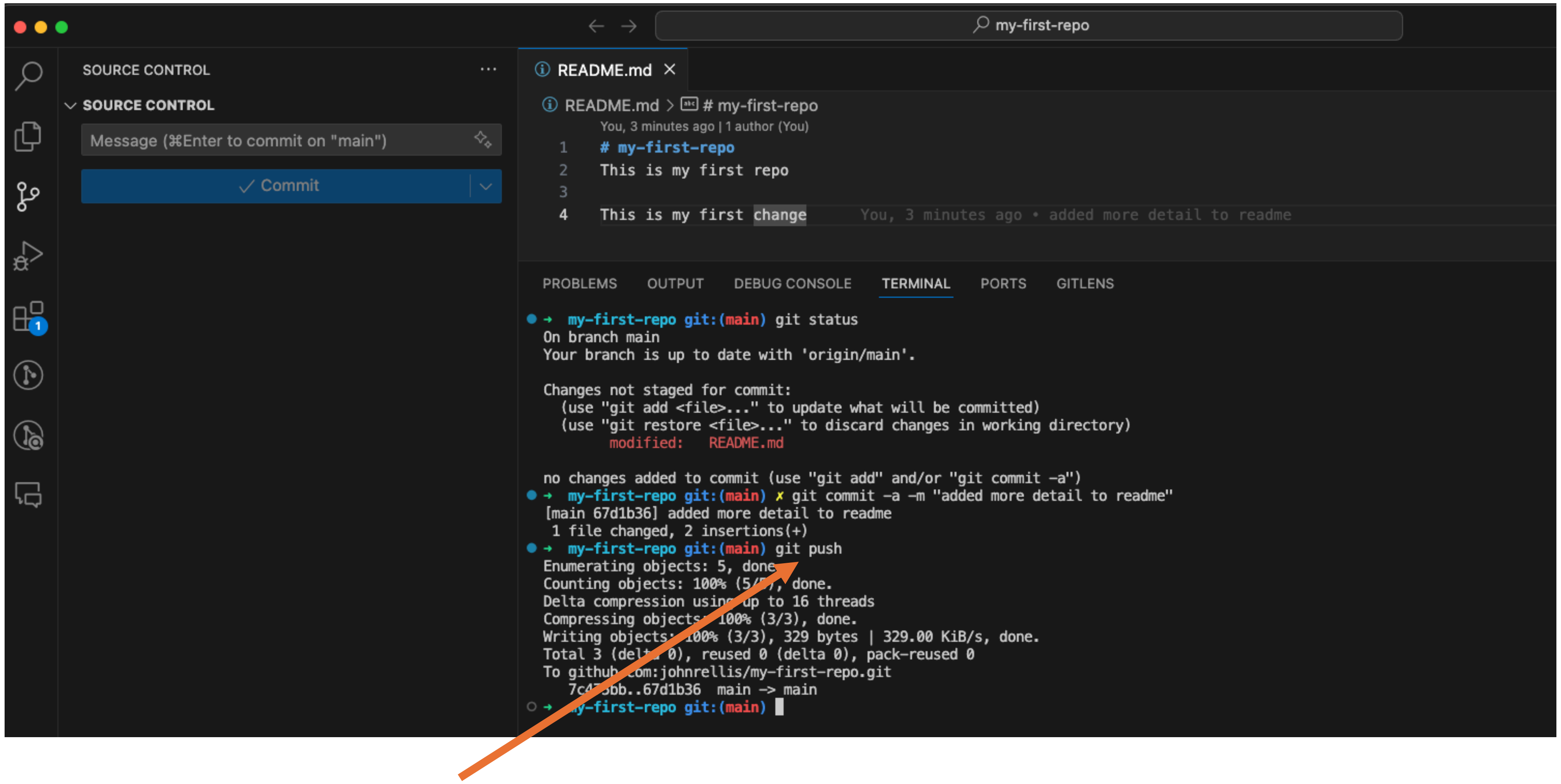
```
● ➜  my-first-repo git:(main) git status
  On branch main
  Your branch is up to date with 'origin/main'.

  Changes not staged for commit:
    (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
          modified:   README.md

  no changes added to commit (use "git add" and/or "git commit -a")
● ➜  my-first-repo git:(main) ✗ git commit -a -m "added more detail to readme"
  [main 67d1b36] added more detail to readme
   1 file changed, 2 insertions(+)
○ ➜  my-first-repo git:(main) 
```

3. Notice we have changes to “sync” – I don’t like this term in git but vs code does

1. git status to see the current status of the repo

2. Commit all changes with a message

Open the directory/repo in VS Code

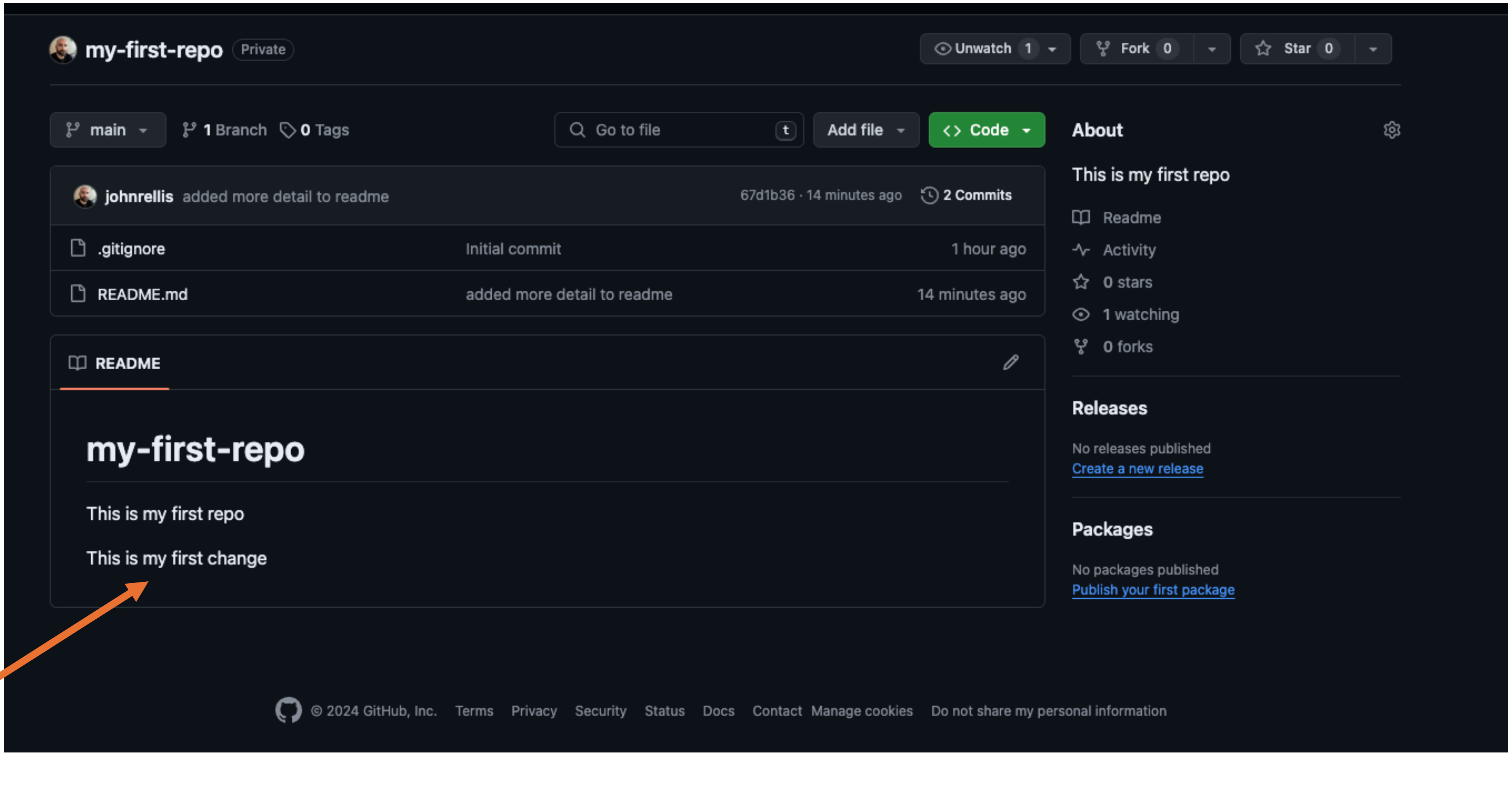


```
# my-first-repo
This is my first repo

This is my first change        You, 3 minutes ago • added more detail to readme
```

```
● → my-first-repo git:(main) git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
● → my-first-repo git:(main) ✗ git commit -a -m "added more detail to readme"
[main 67d1b36] added more detail to readme
 1 file changed, 2 insertions(+)
● → my-first-repo git:(main) git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 329 bytes | 329.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:johnrellis/my-first-repo.git
   7c4[illegible]bb..67d1b36  main -> main
○ → my-first-repo git:(main) █
```

1. We perform a git push to push our changes to our remote

Verify your changes on github



johnrellis added more detail to readme 67d1b36 · 14 minutes ago 2 Commits

.gitignore	Initial commit	1 hour ago
README.md	added more detail to readme	14 minutes ago

my-first-repo

This is my first repo

This is my first change

This is my first repo

Readme
Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

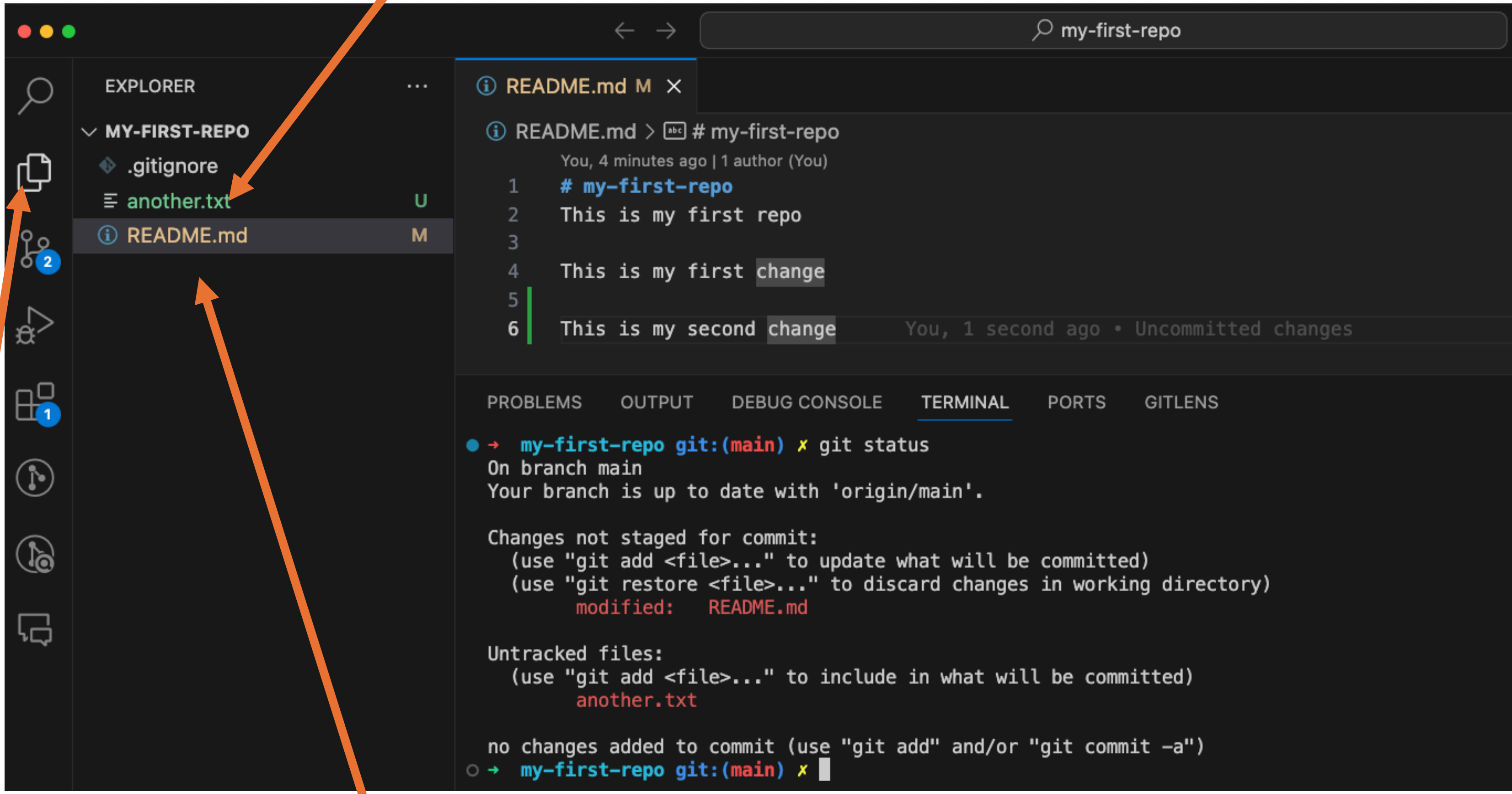
No packages published
Publish your first package

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

Celebrate!

A note on VS Code

- There's a plethora of useful git extensions but when starting out, they can be confusing.
- Personally, I use very little, I work from the explorer and the command line



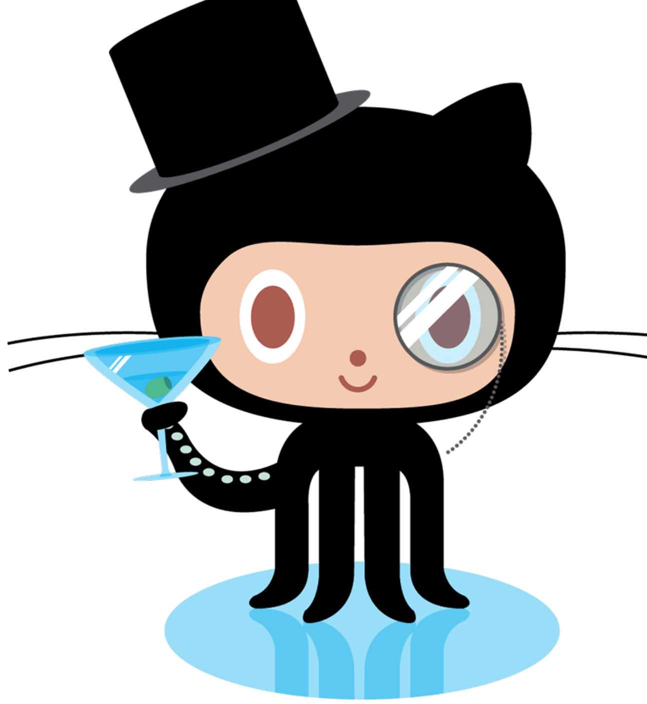
```
# my-first-repo
This is my first repo

This is my first change

This is my second change
```

```
➜ my-first-repo git:(main) ✗ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        another.txt

no changes added to commit (use "git add" and/or "git commit -a")
➜ my-first-repo git:(main) ✗
```

Mini cheatsheet

- git clone <git url> - to clone a repository from git
- git status – check the status of the repo – this is forever in my muscle memory, I do it a lot
- git add –A or git add README.md to stage a file(s)
- git commit –a –m “this adds all new staged files and unstaged mods of existing files”
- git push – will push up to the repo you have cloned

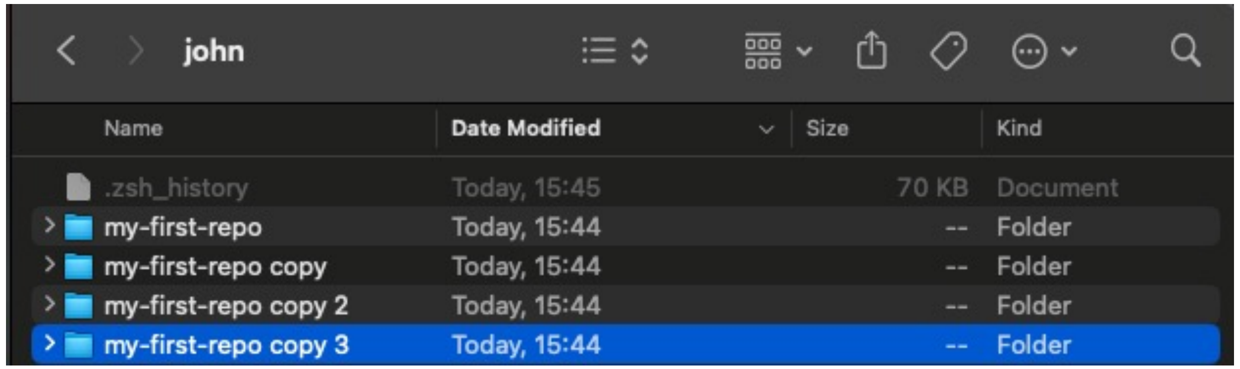
The simple life



- My general workflow is
  - git pull – pull down changes from repo
  - Make a group of related changes
  - git status
  - git add -A (if new files have been added, will do no harm anyway)
  - git commit –a –m “bug-fix: code no longer explodes”
  - git push
  - Rinse and repeat
- There’s nuance you can mix in there, but that can get you going in a kind of unidirectional workflow when working solo.
- It is useful to commit more often than not, as if you make a mistake you can always revert
- Also useful but careful now!
  - git checkout <filename> - undo all changes since last commit on a file
  - git reset --hard – undo all changes since last commit (careful)

Something has gone wrong

- A multitude of things can go wrong when working with projects that have multiple collaborators.
- Take your time, in industry, all changes are peer reviewed so silly mistakes are usually caught
- My secret, if you feel like there's a lot that could go wrong with a git action, make a copy of your repo locally, that way, if you do something you are unsure of in your original, you can just throw it away
- Just be sure to clean up after yourself or you will be confused



Name	Date Modified	Size	Kind
.zsh_history	Today, 15:45	70 KB	Document
my-first-repo	Today, 15:44	--	Folder
my-first-repo copy	Today, 15:44	--	Folder
my-first-repo copy 2	Today, 15:44	--	Folder
my-first-repo copy 3	Today, 15:44	--	Folder

Resources

- https://education.github.com/git-cheat-sheet-education.pdf
- https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet