Introduction to GIT Version Control

James Willis (SciNet)

Nov 21, 2022

Outline

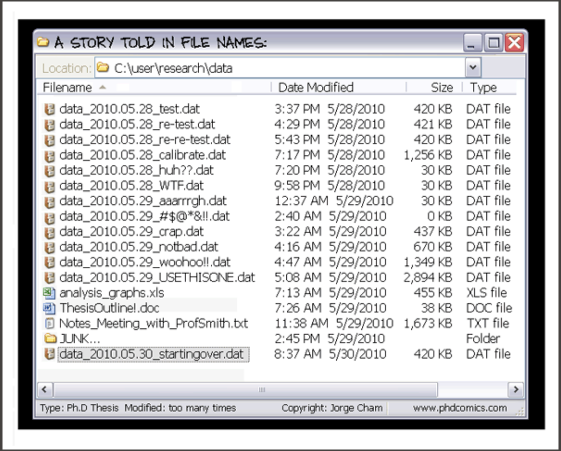
- Why use version control?
- About GIT version control
- GIT commands
  - Hands-on
- Web-based GIT Repos
- GitHub
  - Hands-on

Section 1

Version Control

What is version control?

- Version control is a tool (i.e. a program) for managing changes in a set of files
- It allows you to keep a history of file versions and make them easy to track
- Essentially it takes a “snapshot” of the files in a given moment in time
- Can you think of any examples where you may have used/experience something similar. . . ?



A STORY TOLD IN FILE NAMES:

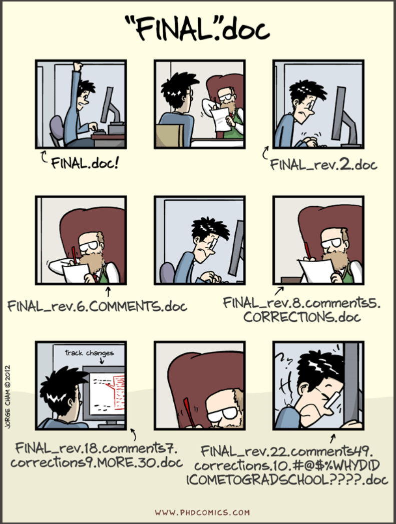
Location: C:\user\research\data

Filename	Date Modified	Size	Type
data_2010.05.28_test.dat	3:37 PM 5/28/2010	420 KB	DAT file
data_2010.05.28_re-test.dat	4:29 PM 5/28/2010	421 KB	DAT file
data_2010.05.28_re-re-test.dat	5:43 PM 5/28/2010	420 KB	DAT file
data_2010.05.28_calibrate.dat	7:17 PM 5/28/2010	1,256 KB	DAT file
data_2010.05.28_huh??.dat	7:20 PM 5/28/2010	30 KB	DAT file
data_2010.05.28_WTF.dat	9:58 PM 5/28/2010	30 KB	DAT file
data_2010.05.29_aaarrrgh.dat	12:37 AM 5/29/2010	30 KB	DAT file
data_2010.05.29_#$@*&!!.dat	2:40 AM 5/29/2010	0 KB	DAT file
data_2010.05.29_crap.dat	3:22 AM 5/29/2010	437 KB	DAT file
data_2010.05.29_notbad.dat	4:16 AM 5/29/2010	670 KB	DAT file
data_2010.05.29_woohoo!!.dat	4:47 AM 5/29/2010	1,349 KB	DAT file
data_2010.05.29_USETHISONE.dat	5:08 AM 5/29/2010	2,894 KB	DAT file
analysis_graphs.xls	7:13 AM 5/29/2010	455 KB	XLS file
ThesisOutline!.doc	7:26 AM 5/29/2010	38 KB	DOC file
Notes_Meeting_with_ProfSmith.txt	11:38 AM 5/29/2010	1,673 KB	TXT file
JUNK...	2:45 PM 5/29/2010		Folder
data_2010.05.30_startingover.dat	8:37 AM 5/30/2010	420 KB	DAT file

Type: Ph.D Thesis Modified: too many times Copyright: Jorge Cham www.phdcomics.com

Why use version control?

- Makes collaboration easier
- Helps you stay organised
- Allows you to keep track of changes without keeping duplicated copies of the same file
- Allows reproducibility
- When something goes wrong, you can back up to the last "working" copy
- It can be used for writing code, writing papers, it is especially powerful for text-based documents
- It is considered a **must** in professional software development


"FINAL".doc
FINAL.doc!
FINAL_rev.2.doc
FINAL_rev.6.COMMENTS.doc
FINAL_rev.8.comments5.CORRECTIONS.doc
track changes
FINAL_rev.18.comments7.corrections9.MORE.30.doc
FINAL_rev.22.comments49.corrections.10.#@$%WHYDID ICOMETOGRADSCHOOL????.doc
JORGE CHAM © 2012
WWW.PHDCOMICS.COM

Examples

You may be familiar with the main features of Version Control already:

- Google Docs/Sheets/Slides
- Overleaf
- Dropbox
- Microsoft Word

These are **not** really Version Control though!

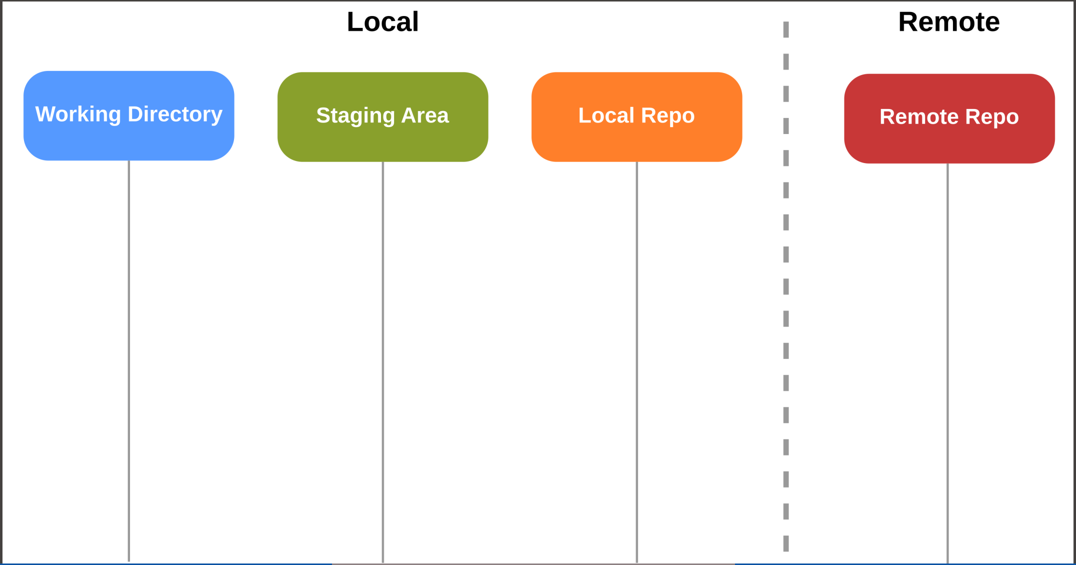
Section 2

GIT

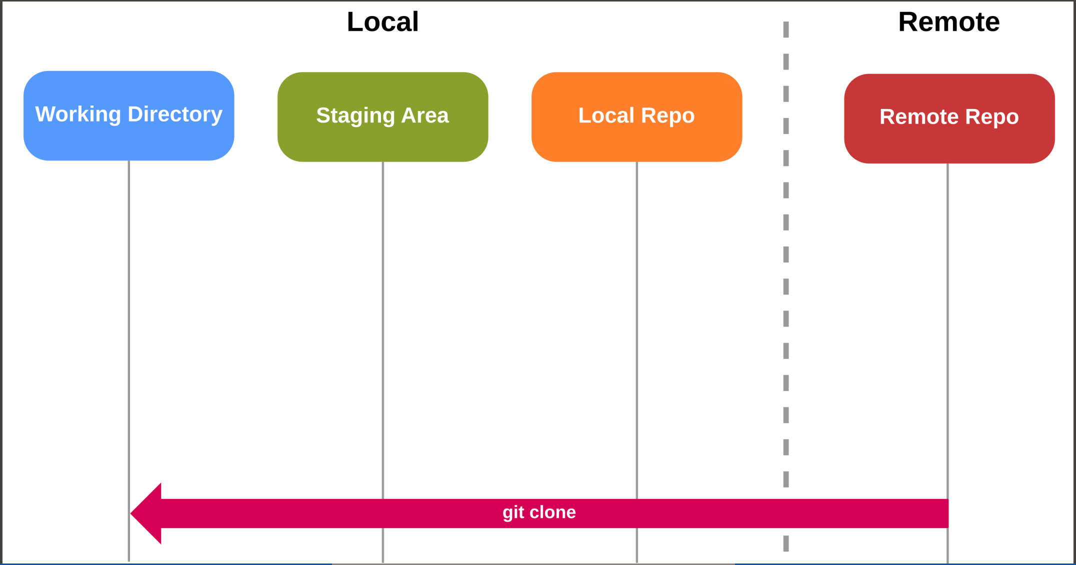
GIT

- Created by *Linus Towalds* in 2005
- What does **GIT** stand for? (https://en.wikipedia.org/wiki/Git#naming)
- There are many types and approaches to version control
- GIT is just one implementation, but it has taken over as the most popular and is used all over the world
- Other implementations include: CVS, SVN, Mercurial, etc. . .
- Some IDEs incorporate VC systems in their GUIs (e.g. Rstudio, Visual Studio, etc. . .)
- And of course, as we will discuss later, there are web-based repositories that allow you to use VC/GIT from within a browser

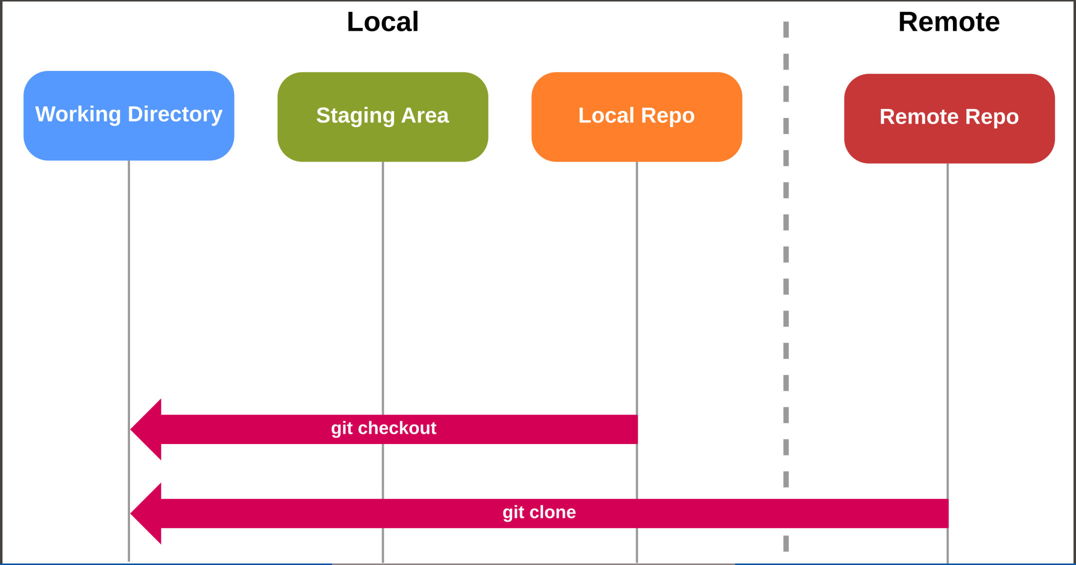
How does GIT work?



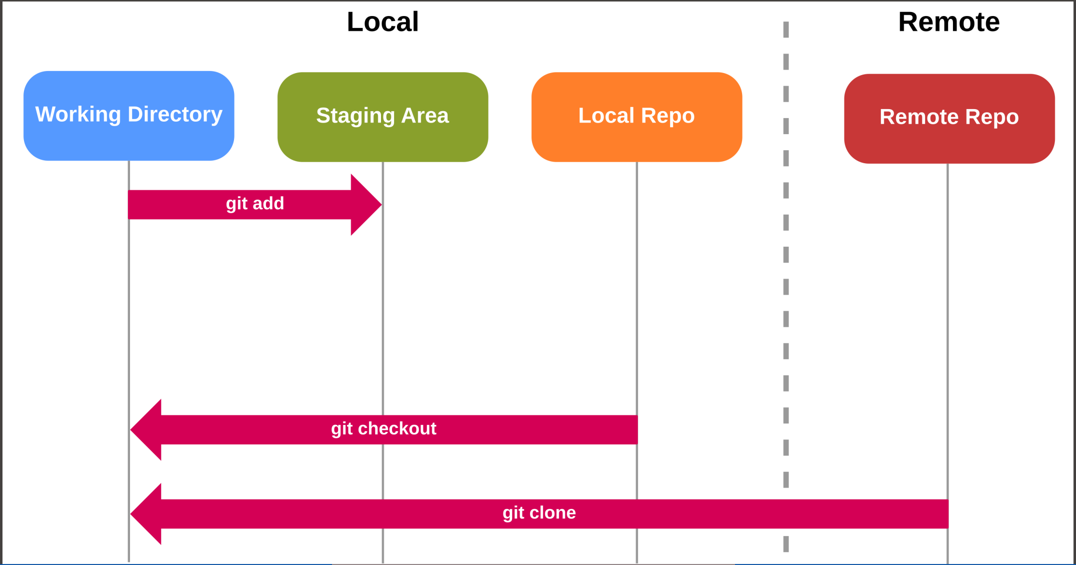
How does GIT work?



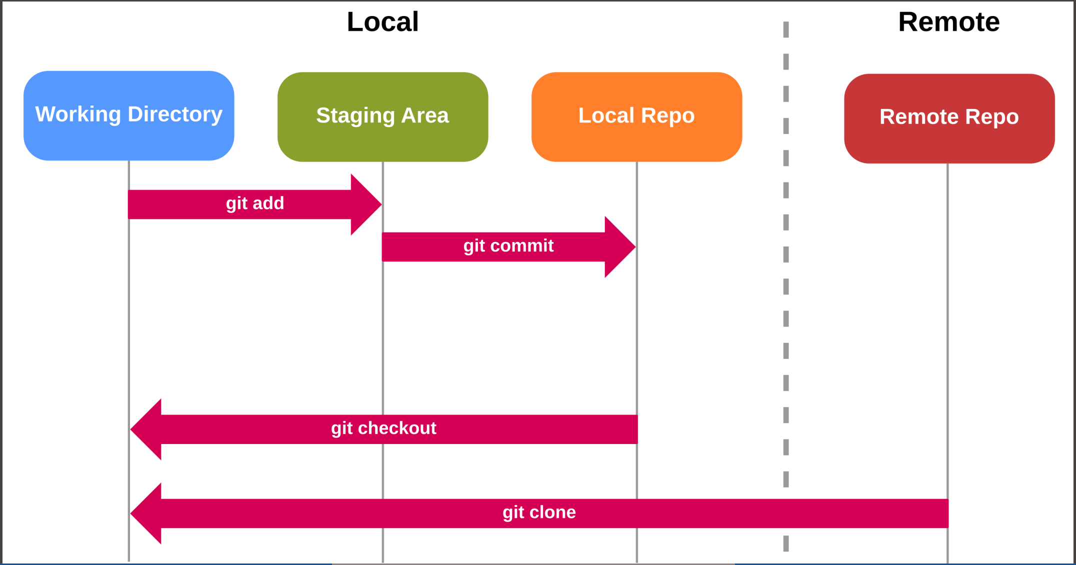
How does GIT work?



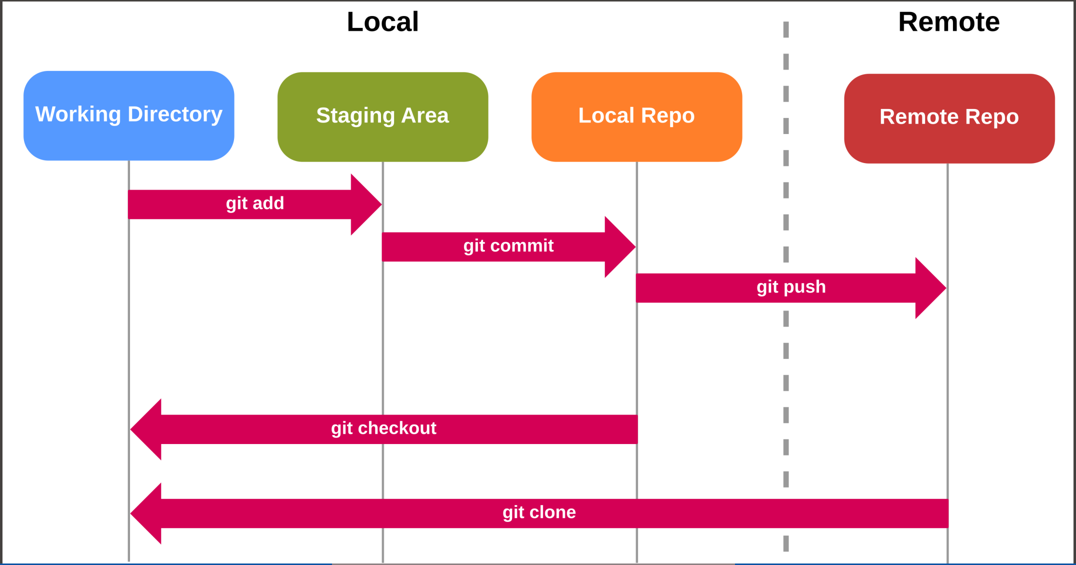
How does GIT work?



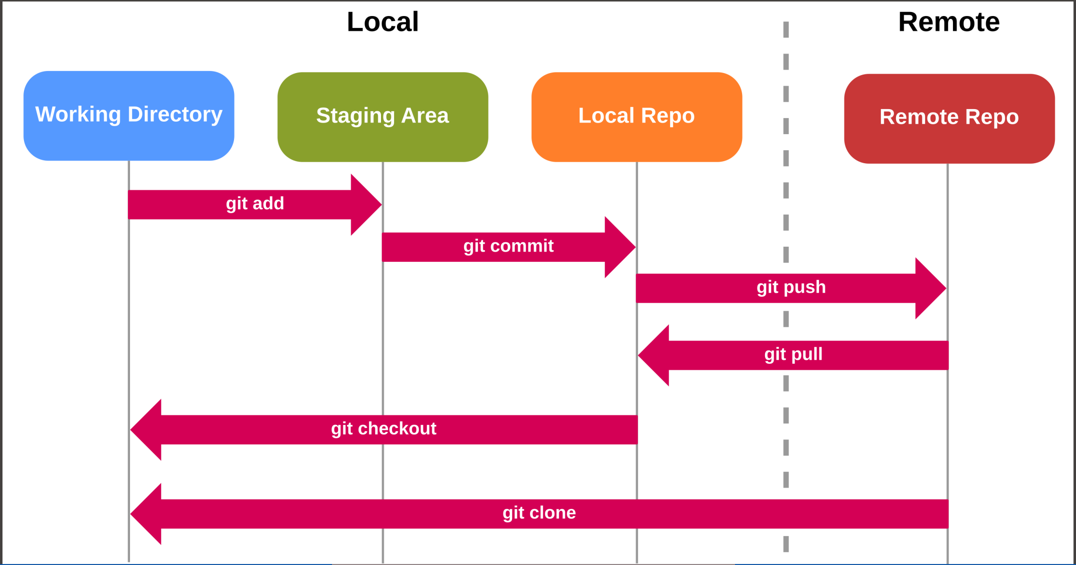
How does GIT work?



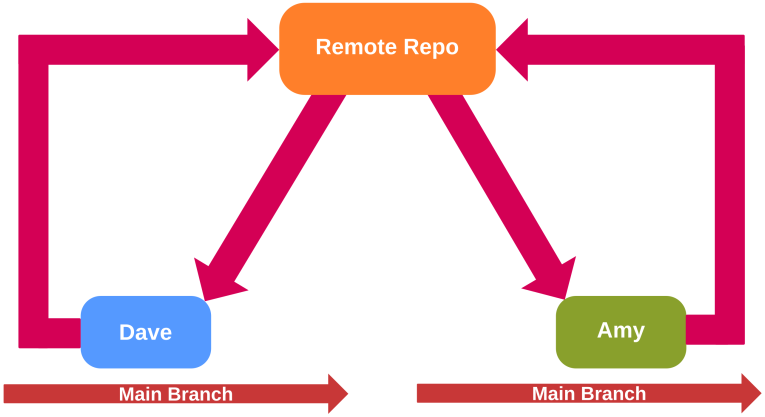
How does GIT work?



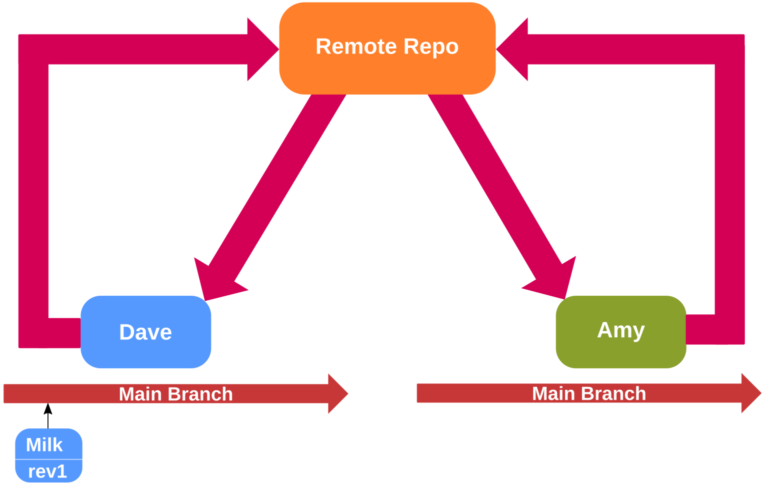
How does GIT work?



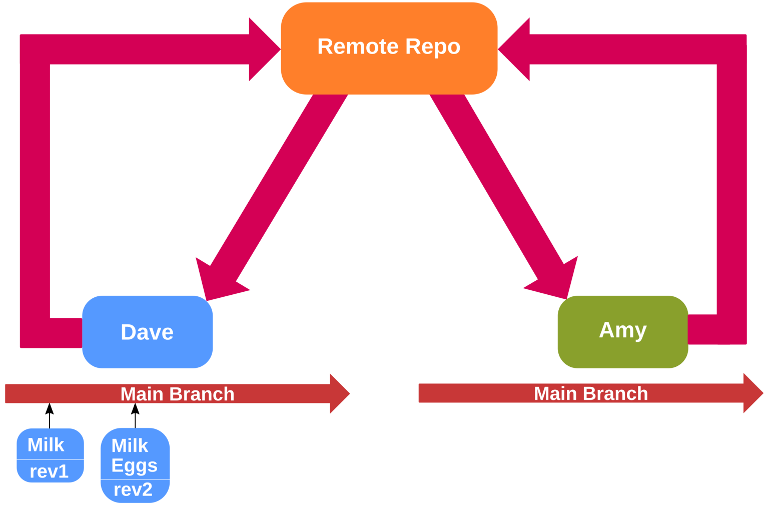
Example



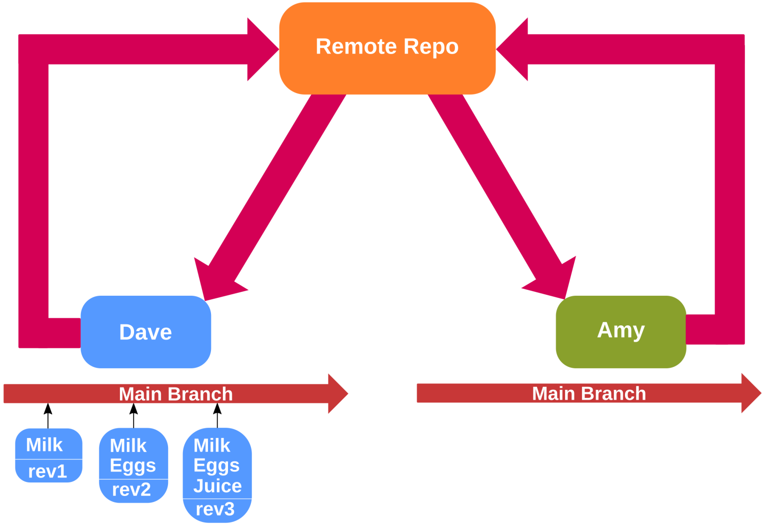
Example



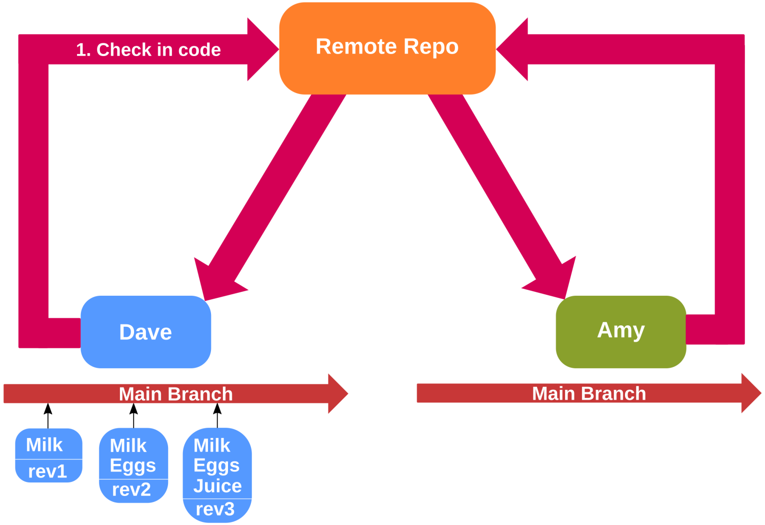
Example


Remote Repo
Dave
Amy
Main Branch
Main Branch
Milk
rev1
Milk
Eggs
rev2

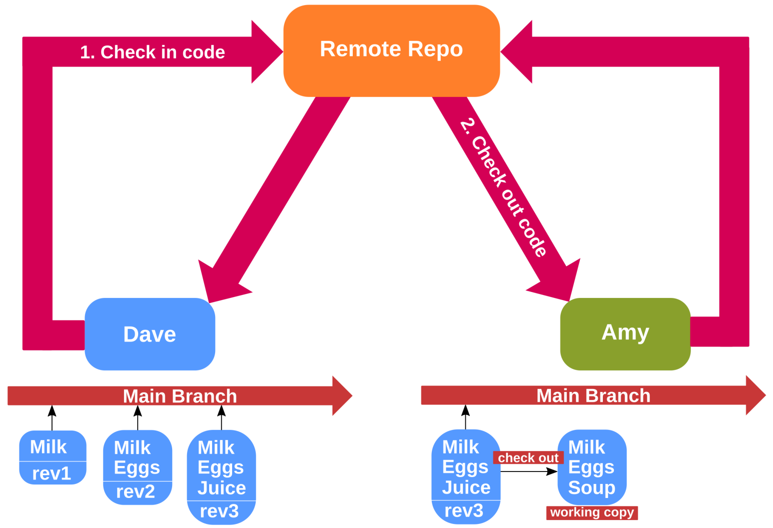
Example


Remote Repo
Dave
Amy
Main Branch
Main Branch
Milk
rev1
Milk
Eggs
rev2
Milk
Eggs
Juice
rev3

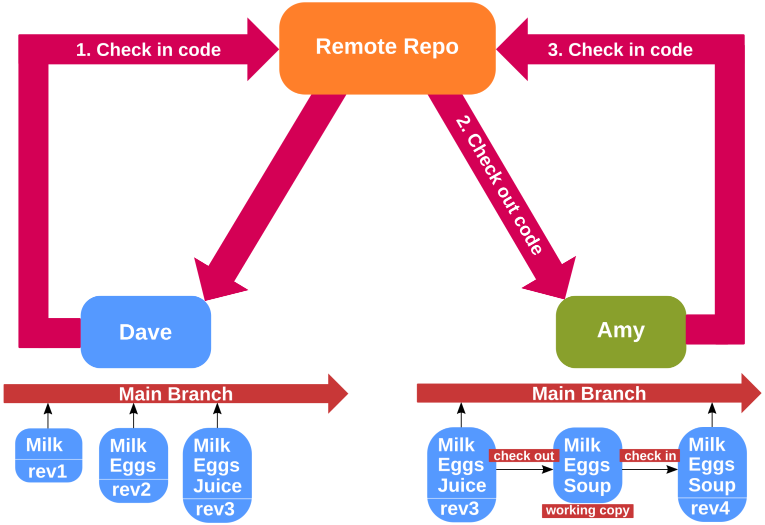
Example


1. Check in code
Remote Repo
Dave
Amy
Main Branch
Main Branch
Milk
rev1
Milk
Eggs
rev2
Milk
Eggs
Juice
rev3

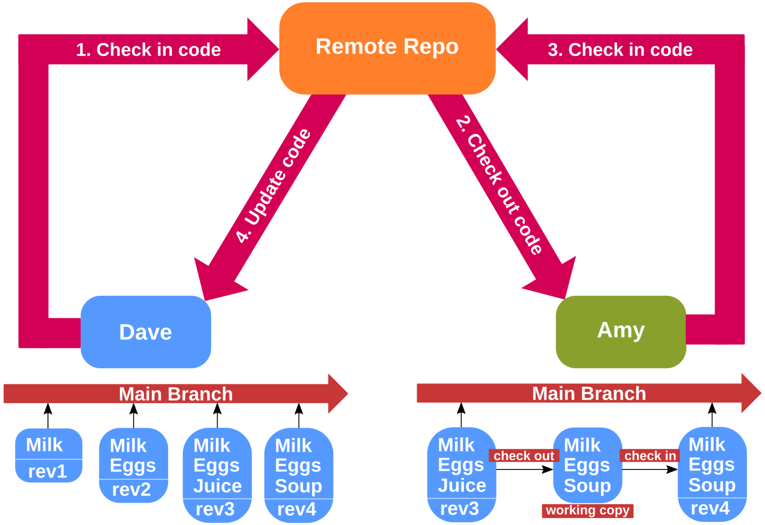
Example


1. Check in code
Remote Repo
2. Check out code
Dave
Amy
Main Branch
Milk
rev1
Milk
Eggs
rev2
Milk
Eggs
Juice
rev3
Main Branch
Milk
Eggs
Juice
rev3
check out
Milk
Eggs
Soup
working copy

Example


Remote Repo
1. Check in code
2. Check out code
3. Check in code
Dave
Amy
Main Branch
Milk
rev1
Milk
Eggs
rev2
Milk
Eggs
Juice
rev3
Main Branch
Milk
Eggs
Juice
rev3
check out
Milk
Eggs
Soup
working copy
check in
Milk
Eggs
Soup
rev4

Example


Remote Repo
1. Check in code
2. Check out code
3. Check in code
4. Update code
Dave
Amy
Main Branch
Milk rev1
Milk Eggs rev2
Milk Eggs Juice rev3
Milk Eggs Soup rev4
Main Branch
Milk Eggs Juice rev3
check out
Milk Eggs Soup
working copy
check in
Milk Eggs Soup rev4

GIT: Terminology

- *Repository*: "A collection of refs together with an object database containing all objects which are reachable from the refs."
- *Commit*: "A single point in the Git history."
- *Checkout*: "The action of updating all or part of the working tree with a tree object or blob from the object database."
- *Branch*: "A branch is used to develop a feature that is merged into the *master* branch upon completion."
- *Conflict*: "When two branches are merged and one branch overwrites changes from the other. All *conflicts* need to be resolved before completing the merge".

GIT: Using a repository

- Step 0: Setup GIT on your computer
- Step 1: Initialise a GIT repo
- Step 2: Commit files to the repo
- Step 3: Edit/Modify/Add new or existing files
- Step 4: Commit changes
- Step 5: Push changes to remote repo
- Step 6: Repeat from Step 2

GIT: Installation

- Install GIT on Linux:

```
sudo apt install git-all
```

- Install GIT on MacOS:

```
git --version
```

(It should prompt you to install if it doesn't already exist)

- Install GIT on Windows by downloading packages from here: https://git-scm.com/download/win
- More information can be found here: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

Section 3

GIT commands

GIT: Creating a repository

- Find a location for your repo and initialise it:

```
laptop:~$ mkdir my-repo
laptop:~$ cd my-repo
laptop:~/my-repo$ git init
Initialized empty Git repository in /home/willis/my-repo/.git/
```

- This creates a `.git` repo in the `my-repo` directory, which contains the repo information:

```
laptop:~/my-repo$ ls -a
.   ..   .git
```

Note: The `-a` option for `ls` shows **all** files, which includes *hidden* files that start with .

GIT: Setting your ID

- The first time you try to use `git` to `commit` something, it might complain that cannot identify you:

```
*** Please tell me who you are.
Run
git config --global user.email "youremail@example.com"
git config --global user.name "FirstName LastName"
To set your account's default identity.
Omit --global to set the identity only in this repository.
fatal: empty indent name (for <(null)>) not allowed
```

- You can also check in advance using:

```
laptop:~$ git config user.name
laptop:~$ git config user.email
```

GIT: Adding files to the repo

Adding files to a repository, requires two steps:

- Step 1: Add files to the *staging* area:

```
laptop:~/my-repo$ echo "some data" > datafile.dat
laptop:~/my-repo$ cp datafile.dat replicated_data.dat
laptop:~/my-repo$ ls
datafile.dat  replicated_data.dat
laptop:~/my-repo$ git add datafile.dat replicated_data.dat
```

- Step 2: Commit files to the repo:

```
laptop:~/my-repo$ git commit datafile.dat replicated_data.dat -m "Adding data from
experiment X."
[master (root-commit) d67dfb5] Adding data from experiment X.
2 files changed, 2 insertions(+)
create mode 100644 datafile.dat
create mode 100644 replicated_data.dat
```

GIT: Comparing file versions

- Suppose we have to update some data and we would like to compare it with the files already in the repo:

```
laptop:~/my-repo$ echo "updated data" >> datafile.dat
laptop:~/my-repo$ git diff datafile.dat
diff --git a/datafile.dat b/datafile.dat
index 4268632..db1d6b5 100644
--- a/datafile.dat
+++ b/datafile.dat
@@ -1 +1,2 @@
 some data
+updated data
laptop:~/my-repo$ git commit datafile.dat -m "Updating data from new experiments."
```

GIT: Logs & recovering file versions

- Look at the history of the repo:

```
laptop:~/my-repo$ git log
commit 5afbe1a660ba831026542e2df9474213eb42237f (HEAD -> master)
Author: willis <james.willis@scinet.utoronto.ca>
Date:   Wed Mar 2 14:22:09 2022 -0500

    Updating data from new experiments.

commit d67dfb567d6d9d92a3a4e0aac1924ab10dda3a61
Author: willis <james.willis@scinet.utoronto.ca>
Date:   Wed Mar 2 12:56:50 2022 -0500

    Adding data from experiment X.
```

- Recover a specific version:

```
laptop:~/my-repo$ git checkout d67dfb56
```

GIT: Removing files from repo

- Delete file:

```
laptop:~/my-repo$ git rm replicated_data.dat
laptop:~/my-repo$ git commit -m "Removed replicated data."
[master 8ee5a01] Removed replicated data.
1 file changed, 1 deletion(-)
delete mode 100644 replicated_data.dat
```

- Note: when you delete a file from the repo like this, it is also deleted from your computer. To remove it from the repo only use the `--cached` option:

```
laptop:~/my-repo$ git rm --cached replicated_data.dat
```

GIT: Reverting changes: `reset`, `checkout` & `revert`

Command	Scope	Common use cases
`git reset`	Commit-level	Discard commits in a private branch or uncommited changes
`git reset`	File-level	Unstage a file
`git checkout`	Commit-level	Switch between branches or inspect old snapshots
`git checkout`	File-level	Discard changes in the working directory
`git revert`	Commit-level	Undo commits in a public branch
`git revert`	File-level	N/A

GIT: Check status of files

- Check the status of files in the local repo:

```
laptop:~/my-repo$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
  new file:   new_file.txt

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:   replicated_data.dat

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

GIT: Full list of commands

- Get a comprehensive list of git commands with `git --help`:

```
These are common Git commands used in various situations:
start a working area (see also: git help tutorial)
   clone             Clone a repository into a new directory
   init              Create an empty Git repository or reinitialize an existing one
work on the current change (see also: git help everyday)
   add               Add file contents to the index
   mv                Move or rename a file, a directory, or a symlink
   restore           Restore working tree files
   rm                Remove files from the working tree and from the index
   sparse-checkout   Initialize and modify the sparse-checkout
examine the history and state (see also: git help revisions)
   bisect            Use binary search to find the commit that introduced a bug
   diff              Show changes between commits, commit and working tree, etc
   grep              Print lines matching a pattern
   log               Show commit logs
   ...
```

GIT: Aliases

- git commands can be quite long to type repeatedly. They can be shortened with aliases
- For example, to shorten `git checkout` to `git co` run:

```
laptop:~$ git config --global alias.co checkout
```

- Useful aliases:

```
laptop:~$ git config --global alias.br branch
laptop:~$ git config --global alias.ci commit
laptop:~$ git config --global alias.st status
laptop:~$ git config --global alias.d difftool
```

Note: all `git` configuration options can be found in your HOME directory in `~/.gitconfig`

Final comments

- It may feel like more work in the short term, but USE IT! It will save you from future headaches
- Commit often!
- Include sensible commit messages
- Do not commit derivative files e.g. log files, executables, compiled modules
- It is useful for different kinds of projects: code development, collaborations, papers etc.
- There are different version control systems: GIT, HG, SVN, CVS

Hands-on

- Install GIT on your local machine
  - `sudo apt install git-all` (Linux)
  - `git --version` (MacOS - It should prompt you to install if it doesn't already exist)
- If that fails, GIT is also installed on Niagara
- Create a local repository
- Add some files
- Experiment with different GIT commands (`git --help` for full list)
- Hints:

```
laptop:~$ git init
laptop:~$ git add file.dat
laptop:~$ git commit file.dat -m "Commit message"
```

Section 4

Web-based GIT Repos

Web-based GIT implementations

- GitHub: https://github.com
- BitBucket: https://bitbucket.org
- GitLab: https://gitlab.com

Section 5

GitHub

Overview

- What is GitHub and why use it?
- Integrating a local repository with GitHub

Activities

- Create a repository in GitHub
- *Push* a repository from your computer to GitHub
- *Pull* a repository from GitHub to your computer
- Create and accept a *pull request*

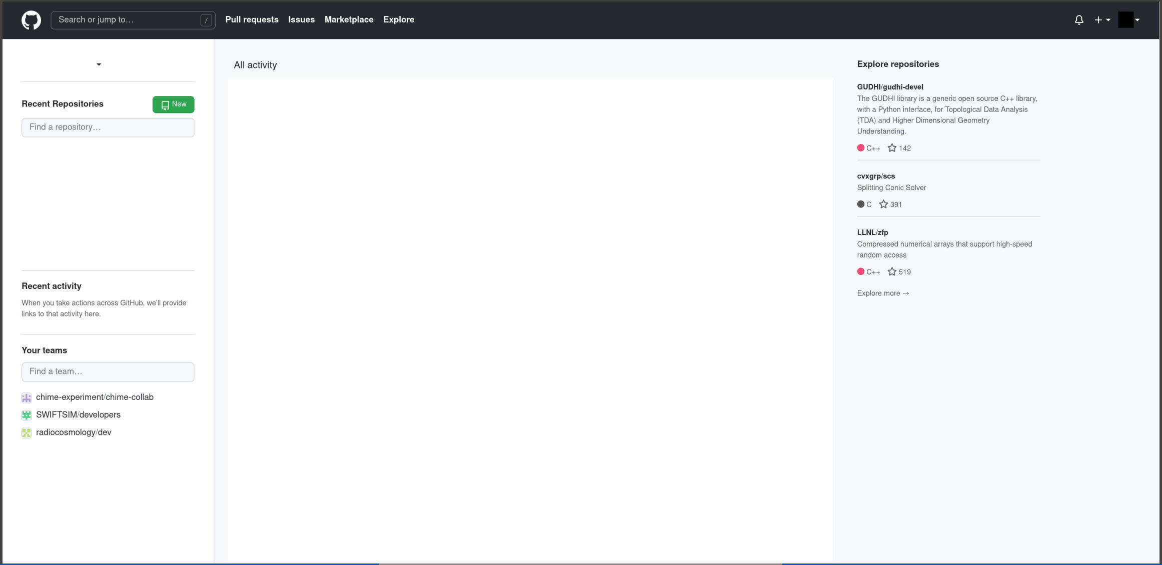
GitHub: Open collaboration

- Git and GitHub *are not the same thing*
- Hosted at github.com
  - Accounts are free
  - Ability to create private repositories
- Heavily used
- Collaborate
- Find code to adapt for you own projects
- Contribute to other code bases

Terminology

- You have a **local** repository on your computer
- GitHub hosts **remote** repositories
- You can **push** from your local repository to a remote repository
- You can **pull** from a remote repository to a local repository
- You can make a **pull request**, in which you ask someone to **pull** your repository into theirs

GitHub: How to create a repo



Recent Repositories

Find a repository…

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

Your teams

Find a team…

chime-experiment/chime-collab

SWIFTSIM/developers

radiocosmology/dev

All activity

Explore repositories

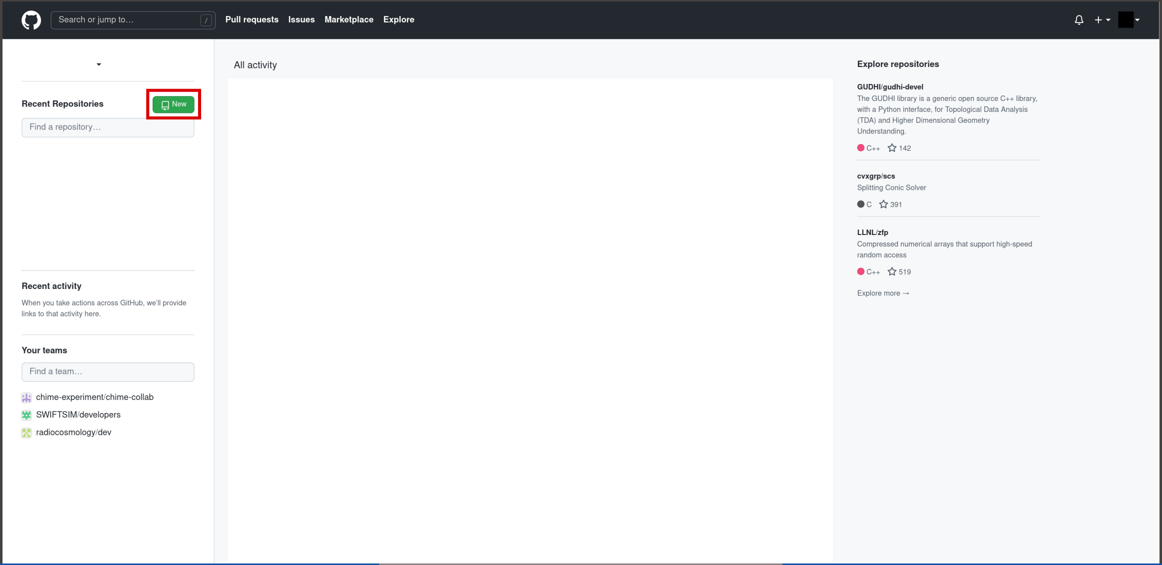
GUDHI/gudhi-devel
The GUDHI library is a generic open source C++ library, with a Python interface, for Topological Data Analysis (TDA) and Higher Dimensional Geometry Understanding.
● C++ ☆ 142

cvxgrp/scs
Splitting Conic Solver
● C ☆ 391

LLNL/zfp
Compressed numerical arrays that support high-speed random access
● C++ ☆ 519

Explore more →

GitHub: How to create a repo



Explore repositories

GUDHI/gudhi-devel
The GUDHI library is a generic open source C++ library, with a Python interface, for Topological Data Analysis (TDA) and Higher Dimensional Geometry Understanding.
● C++ ☆ 142

cvxgrp/scs
Splitting Conic Solver
● C ☆ 391

LLNL/zfp
Compressed numerical arrays that support high-speed random access
● C++ ☆ 519

Explore more →

Recent activity

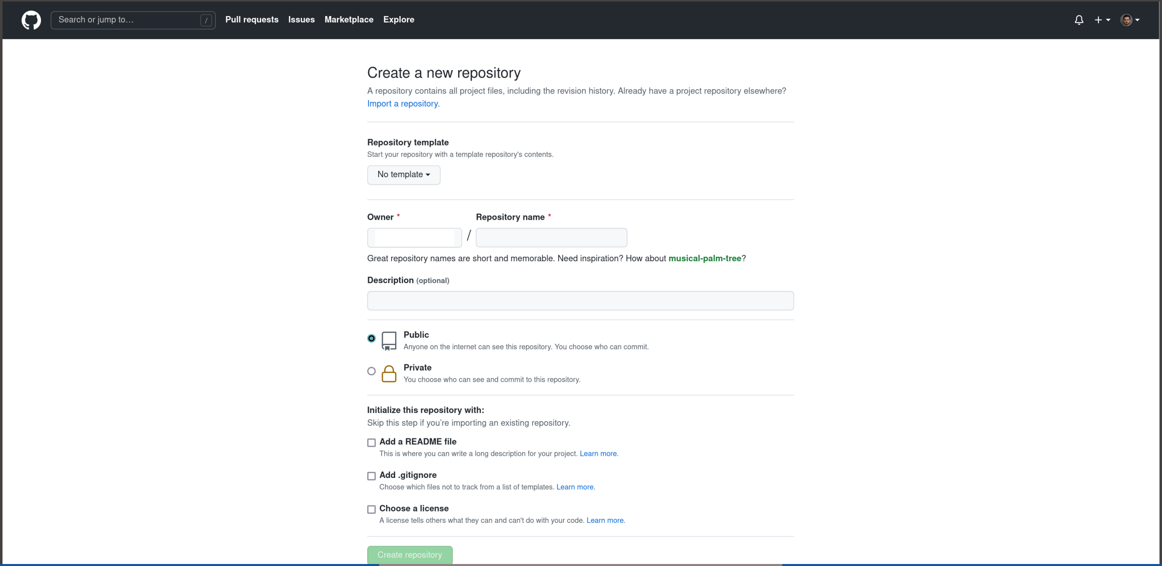
When you take actions across GitHub, we'll provide links to that activity here.

Your teams

Find a team…

chime-experiment/chime-collab
SWIFTSIM/developers
radiocosmology/dev

GitHub: How to create a repo



Create a new repository

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

Repository template

Start your repository with a template repository's contents.

No template

Owner * / **Repository name** *

Great repository names are short and memorable. Need inspiration? How about **musical-palm-tree**?

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:

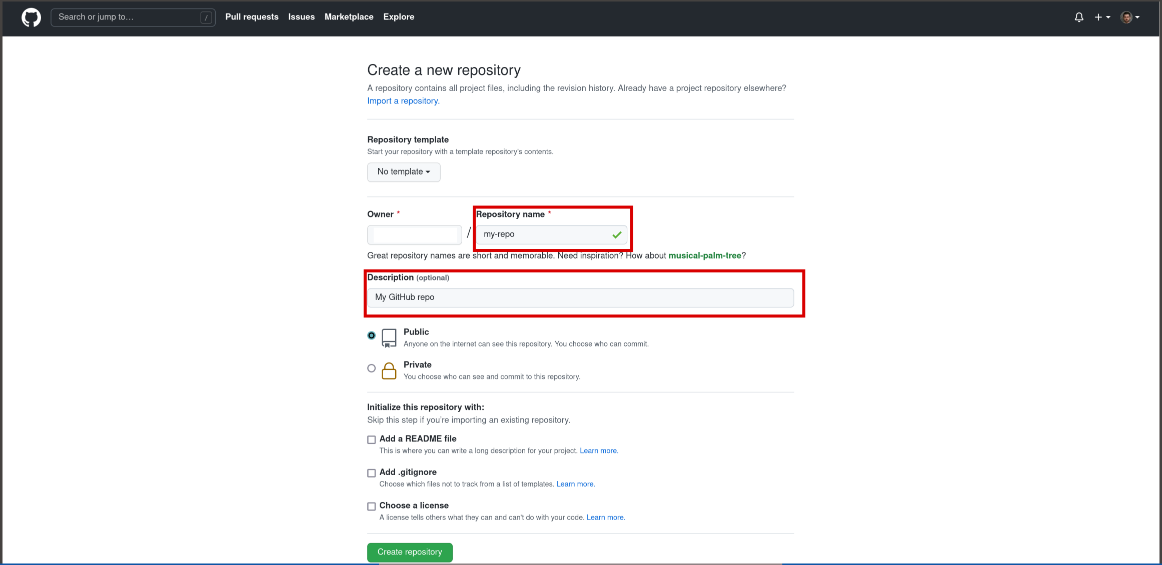
Skip this step if you're importing an existing repository.

☐ **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.

☐ **Choose a license**
A license tells others what they can and can't do with your code. Learn more.

GitHub: How to create a repo



Create a new repository

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

Repository template

Start your repository with a template repository's contents.

No template

Owner *

Repository name *

my-repo

Great repository names are short and memorable. Need inspiration? How about **musical-palm-tree**?

Description (optional)

My GitHub 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:

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

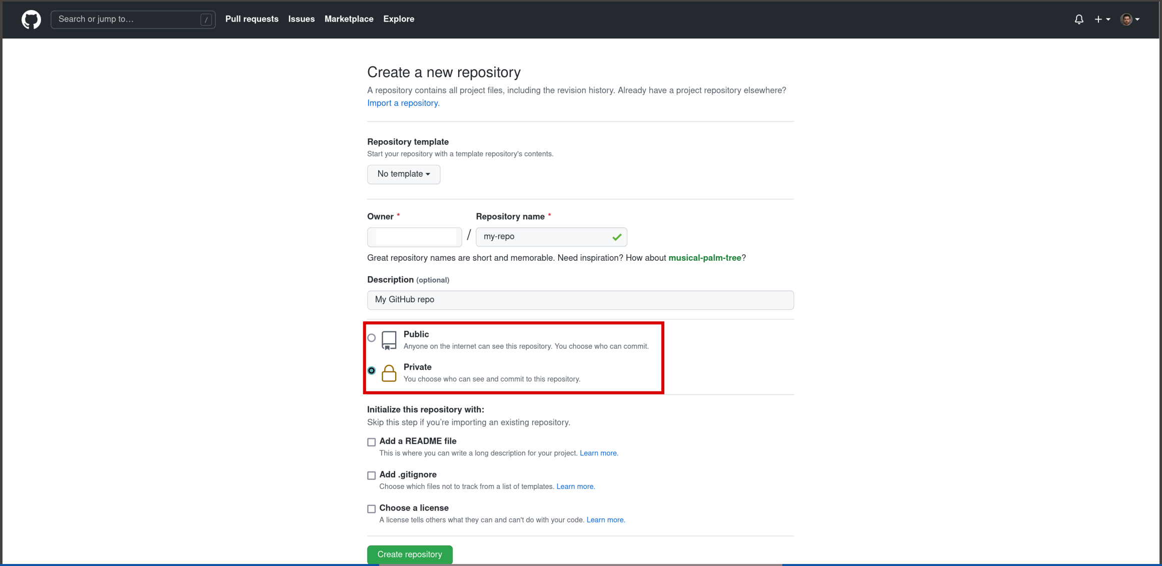
☐ **Add a README file**
This is where you can write a long description for your project. Learn more.

☐ **Add .gitignore**
Choose which files not to track from a list of templates. Learn more.

☐ **Choose a license**
A license tells others what they can and can't do with your code. Learn more.

Create repository

GitHub: How to create a repo



Create a new repository

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

Repository template

Start your repository with a template repository's contents.

No template

Owner * / **Repository name** *

my-repo

Great repository names are short and memorable. Need inspiration? How about **musical-palm-tree**?

Description (optional)

My GitHub 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:

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

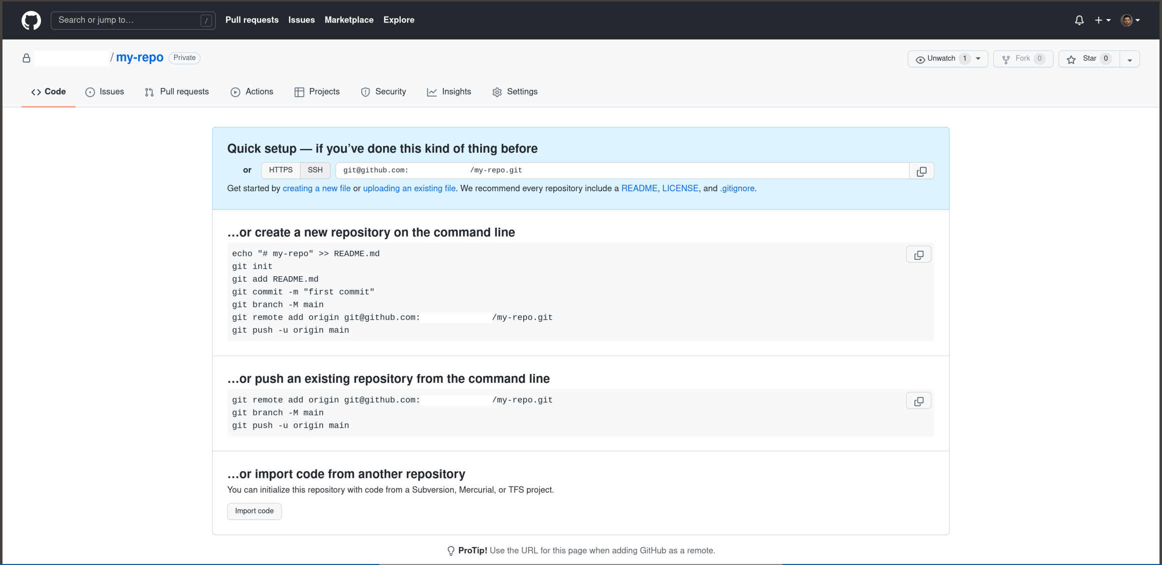
☐ **Add a README file**
This is where you can write a long description for your project. Learn more.

☐ **Add .gitignore**
Choose which files not to track from a list of templates. Learn more.

☐ **Choose a license**
A license tells others what they can and can't do with your code. Learn more.

Create repository

GitHub: How to create a repo



Quick setup — if you've done this kind of thing before

```
git@github.com:            /my-repo.git
```

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

…or create a new repository on the command line

```
echo "# my-repo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:            /my-repo.git
git push -u origin main
```

…or push an existing repository from the command line

```
git remote add origin git@github.com:            /my-repo.git
git branch -M main
git push -u origin main
```

…or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

ProTip! Use the URL for this page when adding GitHub as a remote.

Push a local repo to GitHub

- Let the local repo know where to find the remote GitHub repo:

```
laptop:~$ cd my-repo/
laptop:~/my-repo$ git remote add origin git@github.com:username/my-repo.git
```

Push a local repo to GitHub

- Let the local repo know where to find the remote GitHub repo:

```
laptop:~$ cd my-repo/
laptop:~/my-repo$ git remote add origin git@github.com:username/my-repo.git
```

- Create the main branch and call it `main`:

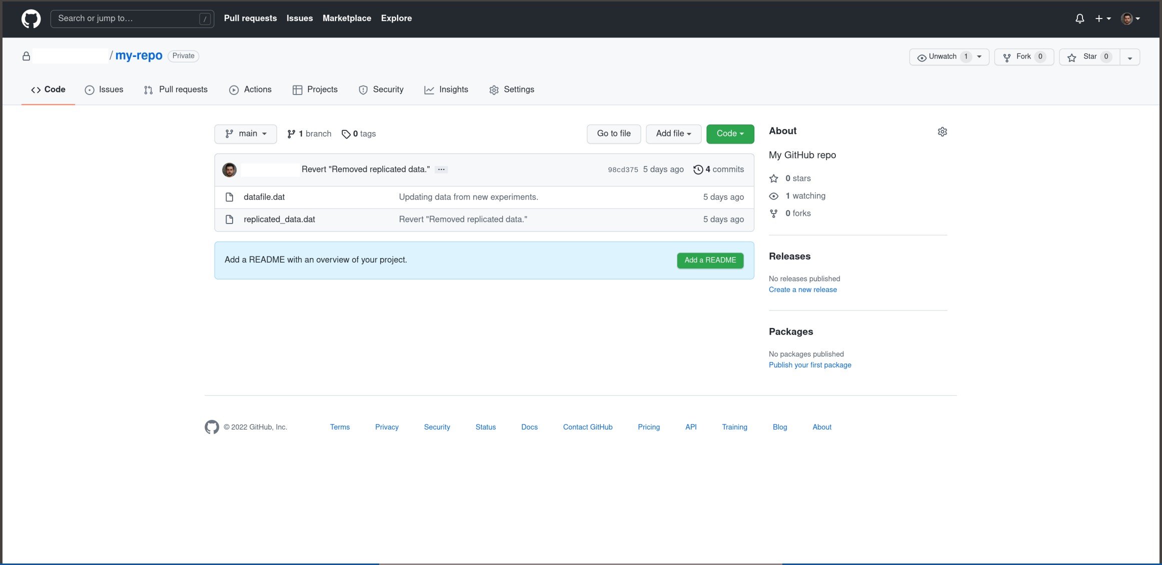
```
laptop:~/my-repo$ git branch -M main
```

Push a local repo to GitHub

- Push your local branch to the remote (origin) GitHub repo:

```
laptop:~/my-repo$ git push -u origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 903 bytes | 903.00 KiB/s, done.
Total 9 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To github.com:username/my-repo.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
```

View repo on GitHub



[redacted] / **my-repo** Private

1 branch 0 tags

[redacted] Revert "Removed replicated data." ··· 98cd375 5 days ago 4 commits

datafile.dat	Updating data from new experiments.	5 days ago
replicated_data.dat	Revert "Removed replicated data."	5 days ago

Add a README with an overview of your project. Add a README

About

My GitHub repo

0 stars
1 watching
0 forks

Releases

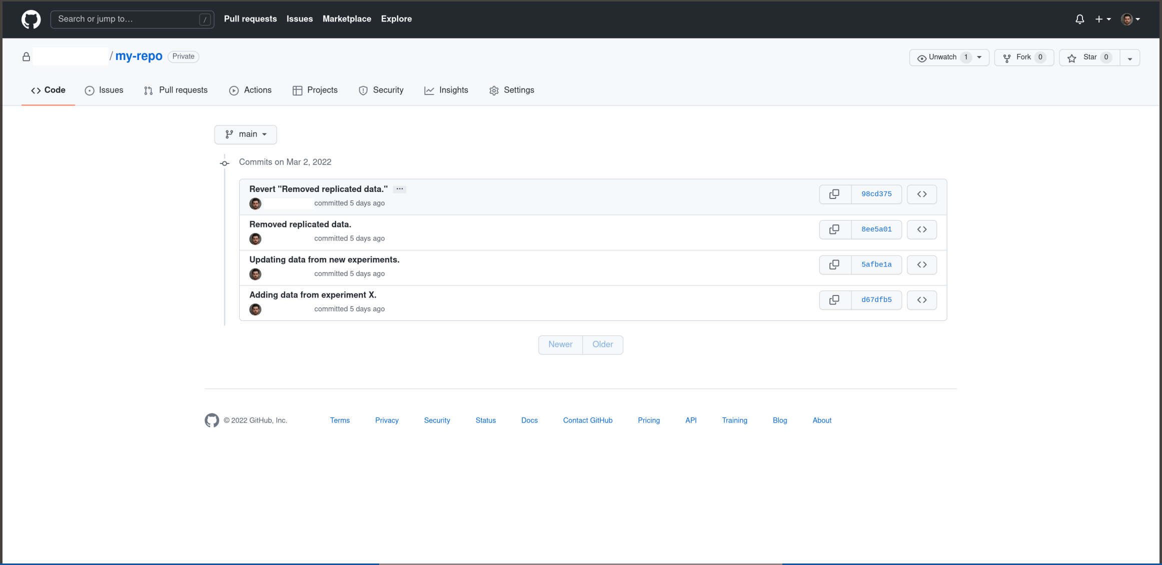
No releases published
Create a new release

Packages

No packages published
Publish your first package

© 2022 GitHub, Inc.

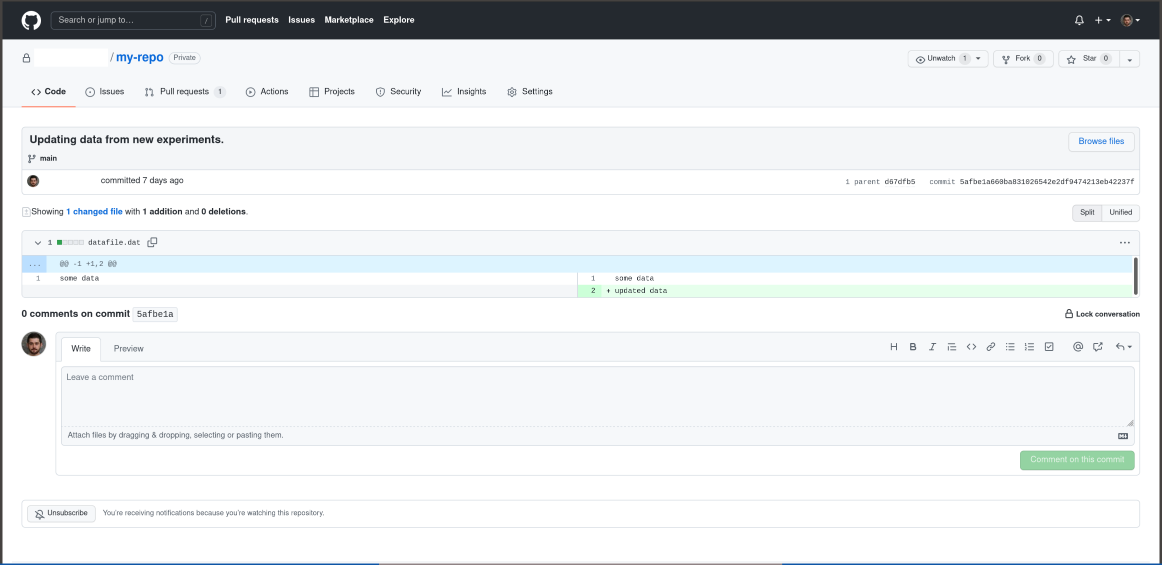
View repo on GitHub



Commits on Mar 2, 2022

Commit		SHA
Revert "Removed replicated data."	committed 5 days ago	98cd375
Removed replicated data.	committed 5 days ago	8ee5a01
Updating data from new experiments.	committed 5 days ago	5afbe1a
Adding data from experiment X.	committed 5 days ago	d67dfb5

View repo on GitHub



Search or jump to... / Pull requests Issues Marketplace Explore

/ my-repo Private

Unwatch 1 · Fork 0 · Star 0

Code · Issues · Pull requests 1 · Actions · Projects · Security · Insights · Settings

Updating data from new experiments.

Browse files

main

committed 7 days ago

1 parent d67dfb5 commit 5afbe1a660ba831026542e2df9474213eb42237f

Showing **1 changed file** with **1 addition** and **0 deletions**.

Split Unified

1 datafile.dat

```
@@ -1 +1,2 @@
1  some data                    1    some data
                               2  + updated data
```

0 comments on commit 5afbe1a

Lock conversation

Write Preview

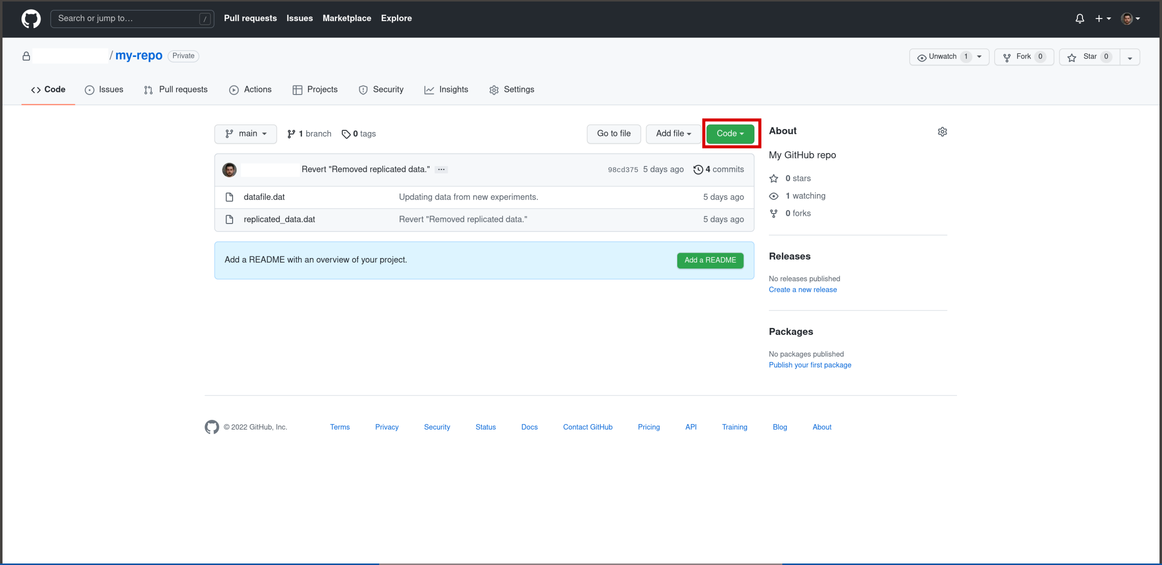
Leave a comment

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

Comment on this commit

Unsubscribe You're receiving notifications because you're watching this repository.

Pull repo from GitHub



Search or jump to… / Pull requests Issues Marketplace Explore

[redacted] / my-repo Private

Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

[redacted] Revert "Removed replicated data." ··· 98cd375 5 days ago 4 commits

datafile.dat	Updating data from new experiments.	5 days ago
replicated_data.dat	Revert "Removed replicated data."	5 days ago

Add a README with an overview of your project. Add a README

About

My GitHub repo

0 stars
1 watching
0 forks

Releases

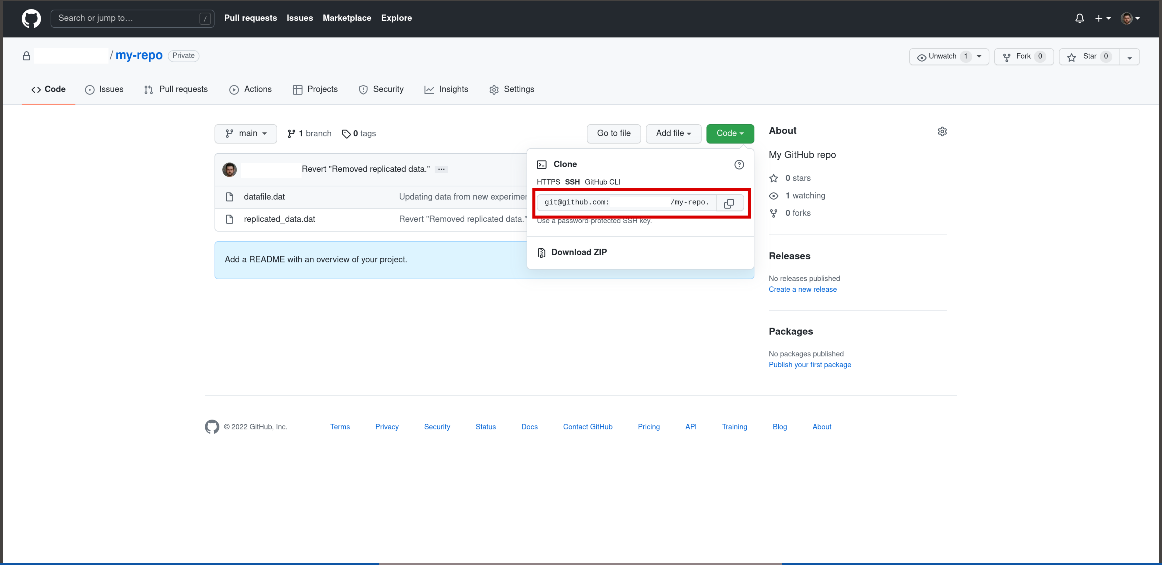
No releases published
Create a new release

Packages

No packages published
Publish your first package

© 2022 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

Pull repo from GitHub



```
git@github.com:          /my-repo.
```

Pull repo from GitHub

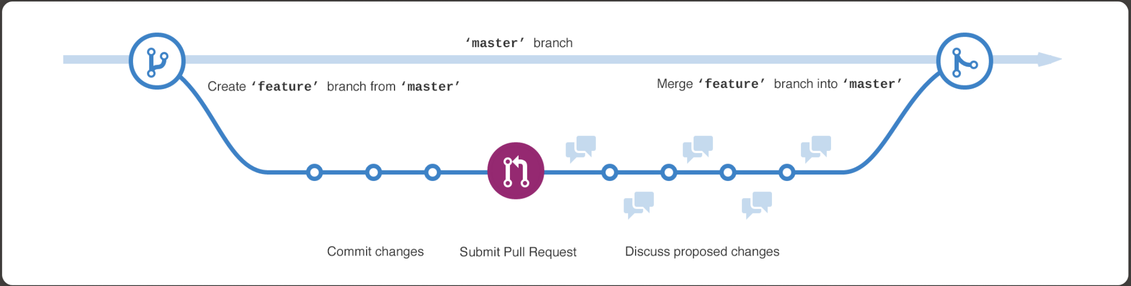
- Pull `my-repo` from GitHub onto *Niagara*:

```
laptop:~$ ssh -A niagara
user@nia-login02:~$ git clone git@github.com:username/my-repo.git
Cloning into 'my-repo'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 9 (delta 1), reused 9 (delta 1), pack-reused 0
Receiving objects: 100% (9/9), done.
Resolving deltas: 100% (1/1), done.
```

GitHub: Pull requests

- Pull requests are a way to *merge* changes from a new branch into the main branch
- They allow teams to review and either accept or reject new changes
- Powerful tool to help prevent new changes from breaking old code
- Can also run regression tests in GitHub CI (Continuous Integration)
- More info on GitHub CI here: https://docs.github.com/en/actions/automating-builds-and-tests/about-continuous-integration

GitHub: Pull requests


'master' branch
Create 'feature' branch from 'master'
Merge 'feature' branch into 'master'
Commit changes
Submit Pull Request
Discuss proposed changes

Source: https://uoft-oss.github.io/git-workflow/

GitHub: Create a pull request

- Create a new branch and add some changes locally:

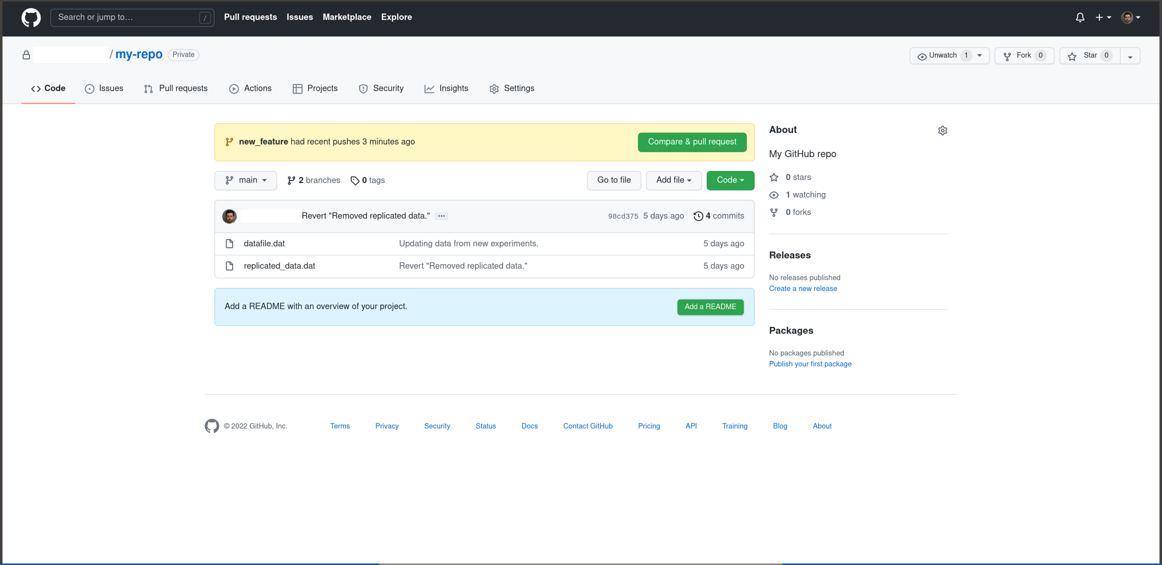
```
laptop:~/my-repo$ git checkout -b new_feature
Switched to a new branch 'new_feature'
laptop:~/my-repo$ git add hello.c
laptop:~/my-repo$ git commit hello.c -m "Hello world program."
[new_feature f3a4091] Hello world program.
1 file changed, 7 insertions(+)
create mode 100644 hello.c
laptop:~/my-repo$ git commit datafile.dat -m "Fixed error."
[new_feature f2b6fe3] Fixed error.
1 file changed, 2 insertions(+), 1 deletion(-)
```

GitHub: Create a pull request

- Push new branch to GitHub:

```
laptop:~/my-repo$ git push
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 16 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 703 bytes | 703.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'new_feature' on GitHub by visiting:
remote:      https://github.com/username/my-repo/pull/new/new_feature
remote:
To github.com:username/my-repo.git
 * [new branch]      new_feature -> new_feature
```

GitHub: Create a pull request



new_feature had recent pushes 3 minutes ago

Revert "Removed replicated data." 98cd375 5 days ago 4 commits

datafile.dat	Updating data from new experiments.	5 days ago
replicated_data.dat	Revert "Removed replicated data."	5 days ago

Add a README with an overview of your project.

About

My GitHub repo

0 stars
1 watching
0 forks

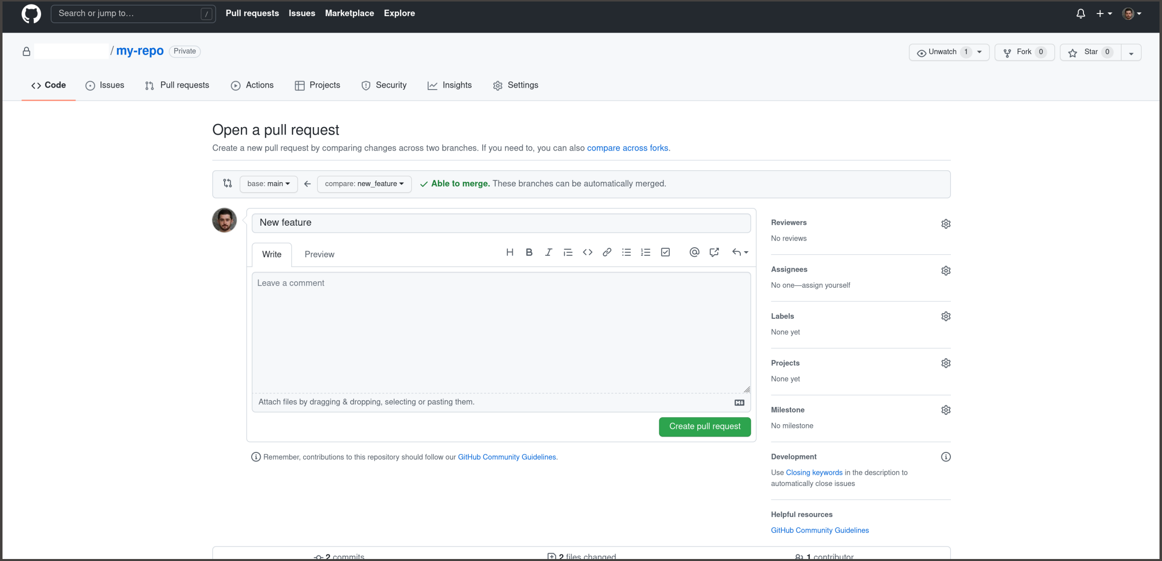
Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

GitHub: Create a 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: new_feature ✓ **Able to merge.** These branches can be automatically merged.

New feature

Leave a comment

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

Remember, contributions to this repository should follow our GitHub Community Guidelines.

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

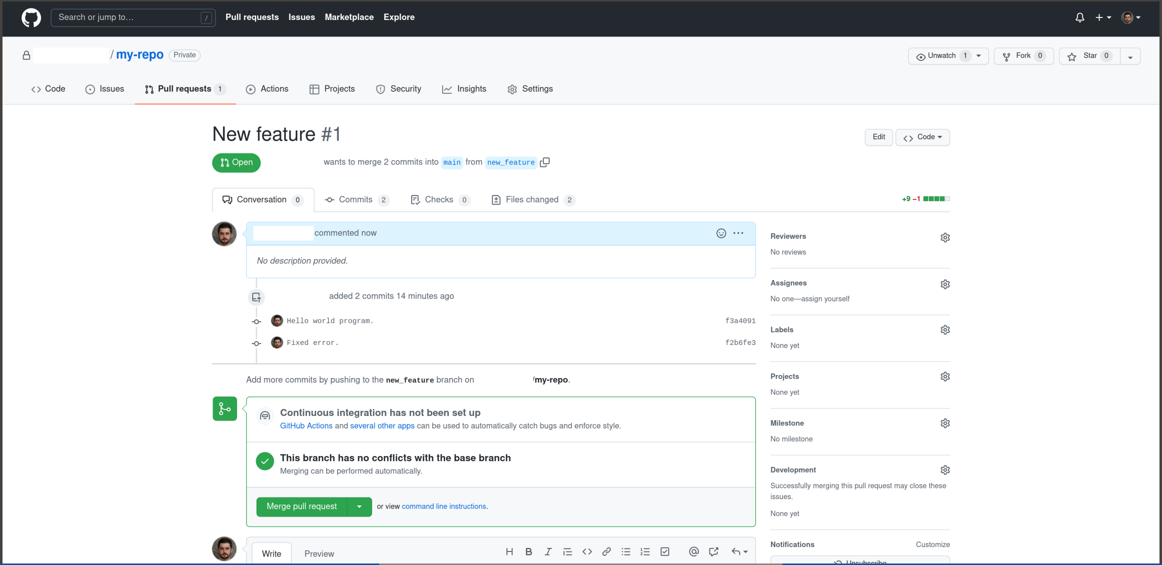
Milestone
No milestone

Development
Use Closing keywords in the description to automatically close issues

Helpful resources
GitHub Community Guidelines

2 commits   2 files changed   1 contributor

GitHub: Create a pull request



Search or jump to... Pull requests Issues Marketplace Explore

/ my-repo Private

Unwatch 1 Fork 0 Star 0

Code Issues Pull requests 1 Actions Projects Security Insights Settings

New feature #1

Edit Code

Open wants to merge 2 commits into `main` from `new_feature`

Conversation 0 Commits 2 Checks 0 Files changed 2 +9 −1

commented now

No description provided.

added 2 commits 14 minutes ago

```
Hello world program.                f3a4091
Fixed error.                        f2b6fe3
```

Add more commits by pushing to the **`new_feature`** branch on /**my-repo**.

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 or view command line instructions.

Write Preview

Reviewers: No reviews

Assignees: No one—assign yourself

Labels: None yet

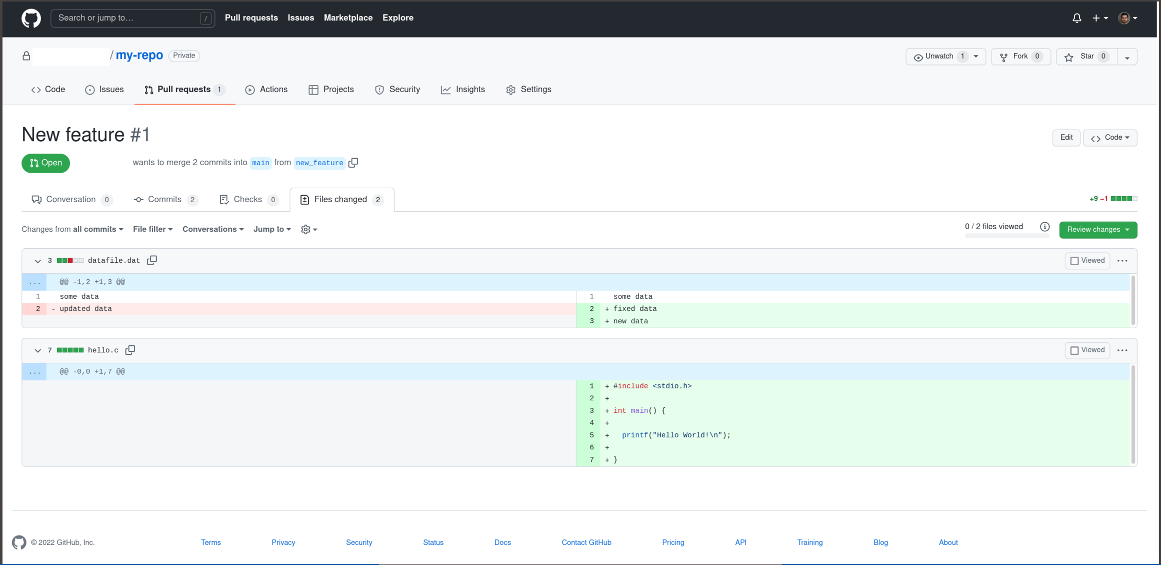
Projects: None yet

Milestone: No milestone

Development: Successfully merging this pull request may close these issues. None yet

Notifications Customize

GitHub: Create a pull request



[redacted] / my-repo Private

New feature #1

Open — wants to merge 2 commits into `main` from `new_feature`

Conversation 0 · Commits 2 · Checks 0 · Files changed 2 — +9 −1

Changes from all commits · File filter · Conversations · Jump to — 0 / 2 files viewed — Review changes

3 datafile.dat

```
@@ -1,2 +1,3 @@
  some data                 some data
- updated data            + fixed data
                          + new data
```

7 hello.c

```
@@ -0,0 +1,7 @@
+ #include <stdio.h>
+
+ int main() {
+
+   printf("Hello World!\n");
+
+ }
```

Hands-on

- Create an account on GitHub
- Add an SSH key to your GitHub account (https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent)
- Create a repository on GitHub
- *Push* a repository from your computer to GitHub:

```
laptop:~/my-repo$ git remote add origin git@github.com:username/my-repo.git
laptop:~/my-repo$ git branch -M main
laptop:~/my-repo$ git push -u origin main
```

- *Pull* your repository from GitHub to your computer:

```
laptop:~$ git clone git@github.com:username/my-repo.git
```

- Create and accept a *pull request*

Further information

Support

Questions? Need help?

Don't be afraid to contact us! We are here to help.

- Email to **support@scinet.utoronto.ca** or to **niagara@computecanada.ca**
- Contact me directly at: **james.willis@scinet.utoronto.ca**.