Version Control

Topics covered

- Disclaimer
- Application to CS425
- What is version control?
- Why version control?
- Version Control Models
- Git Overview
- Git Workflow
- Git Tools
- Contributing to Open Source
- Questions/Comments

Disclaimer

- This course will require the use of Git through GitHub
- We can’t cover everything. Use tutorials to help answer questions.
- Git tutorials and training
  - https://www.atlassian.com/git
  - https://www.coursera.org/learn/version-control-with-git
  - Many organizations offer training as part of employment

Why does this apply to CS425?

- As a part of Project Part 3 deliverables, each team must have a functioning **public** repository on GitHub
  - Your database can be private (if your project has one)
  - Code under an NDA can be kept private

- Add the **public** repository link to your P3 assignment. That’s it!

- Please note that the teaching team will, if necessary, look at the activity in the repository to decide on certain aspects of grading

- This task should take you only 10-20 minutes at most. If you require help, please attend one of our office hours and we will walk you through it.

What is version control?

- The process of tracking and managing changes to software source code
  - Also known as source control

- Essentially, you’re storing your local changes to a remote repository
  - Do not store code on usb drives or Google Drive

- Crucial to software teams
  - Contains loads of software tools that make cooperative programming much easier

- Allows developers to essentially “undo” a mistake

Why version control?

- Accountability
    - Who is contributing to the project?
    - Who is responsible for a check-in (broken code, not following best practices)
- Ownership
    - Finding the creator of an old piece of code for help
    - Getting credit for your work, even years later
- Deployment Pipelines
    - Have a stable release branch that is not used for development
- Industry Practices
    - Version control history can be part of performance reviews
    - “Rolling back” to an old version of the code can help diagnose and fix errors

Why should you care?

Virtually all forms of employment use it

- If they don’t use version control, make them use it or find employment elsewhere

It promotes a group dynamic

- How else would you code with a group of 7+ people?

Everyone makes mistakes

- Ever had a piece of code that was working, then it just didn’t?

Version control models

- Centralized Version Control System (CVCS)
  - The repository is held only on a central server
  - Code is checked into the central repository directly
  - Pros: More administrative powers & control over users and access, smaller local storage, easier to understand
  - Cons: Central point of failure, dependent on connection to central repository
  - Example: Perforce, StarTeam

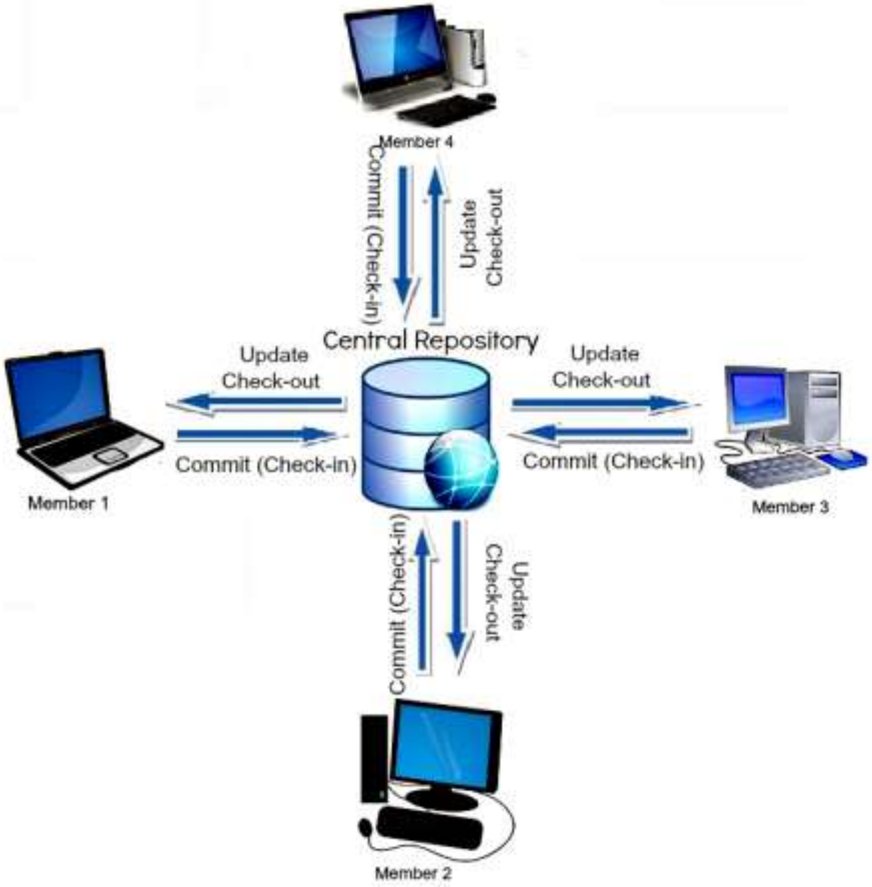


Image Source: https://scmquest.com/centralized-vs-distributed-version-control-systems/

Version control models

- Distributed Version Control System (DVCS)
  - The complete repository is mirrored on every developer’s system
  - Code is checked into the local repository then pushed to the central repository
  - Pros: Enables working offline, comparatively faster, every user has a repository backup
  - Cons: Higher storage requirements, proprietary code leaks more likely
  - Example: Git

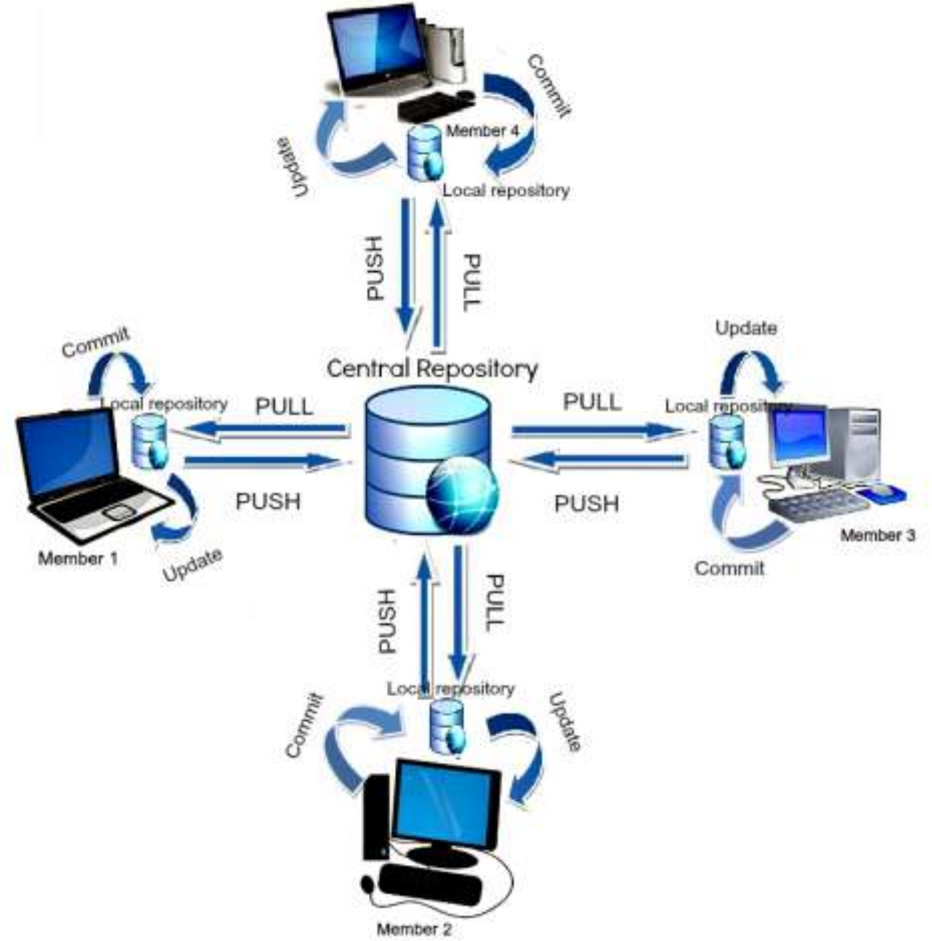
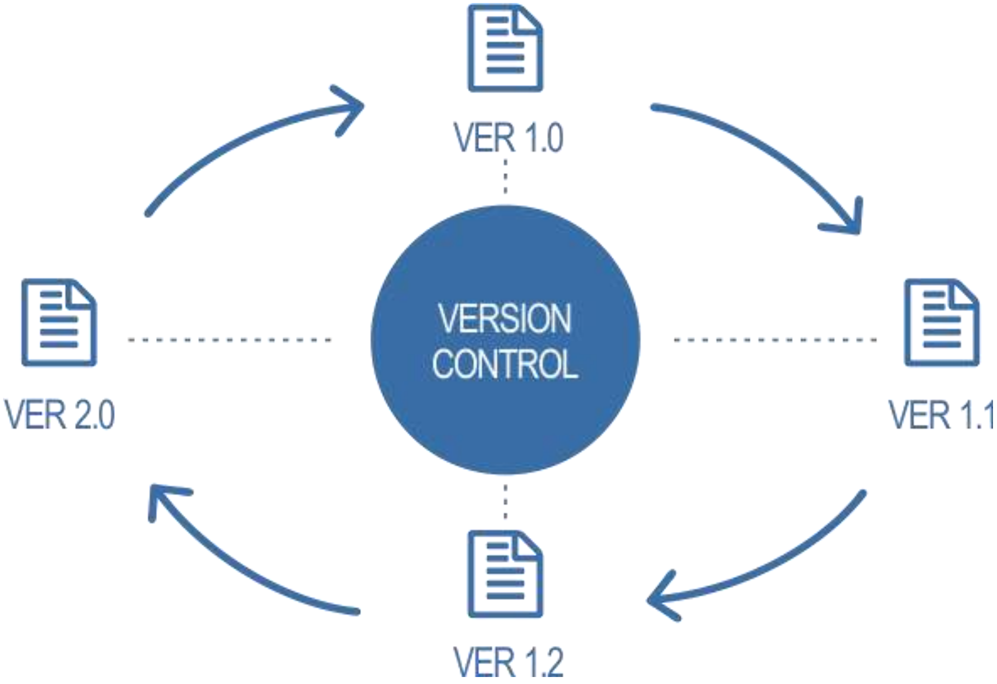


Image Source: https://scmquest.com/centralized-vs-distributed-version-control-systems/

Git Overview: What is Git?

- The most commonly used version control system in the world
- It is the standard in which all version control systems follow
  - Team Foundation Server
  - Bitbucket
  - Apache Subversion
- Git contains its own set of commands, much like linux commands
- It can be a bit confusing at first, but it quickly becomes easier



Git Overview: Git vs GitHub

Git is the version control system itself

GitHub is a hosting service for Git repositories

Git Workflow: Check In & Check Out

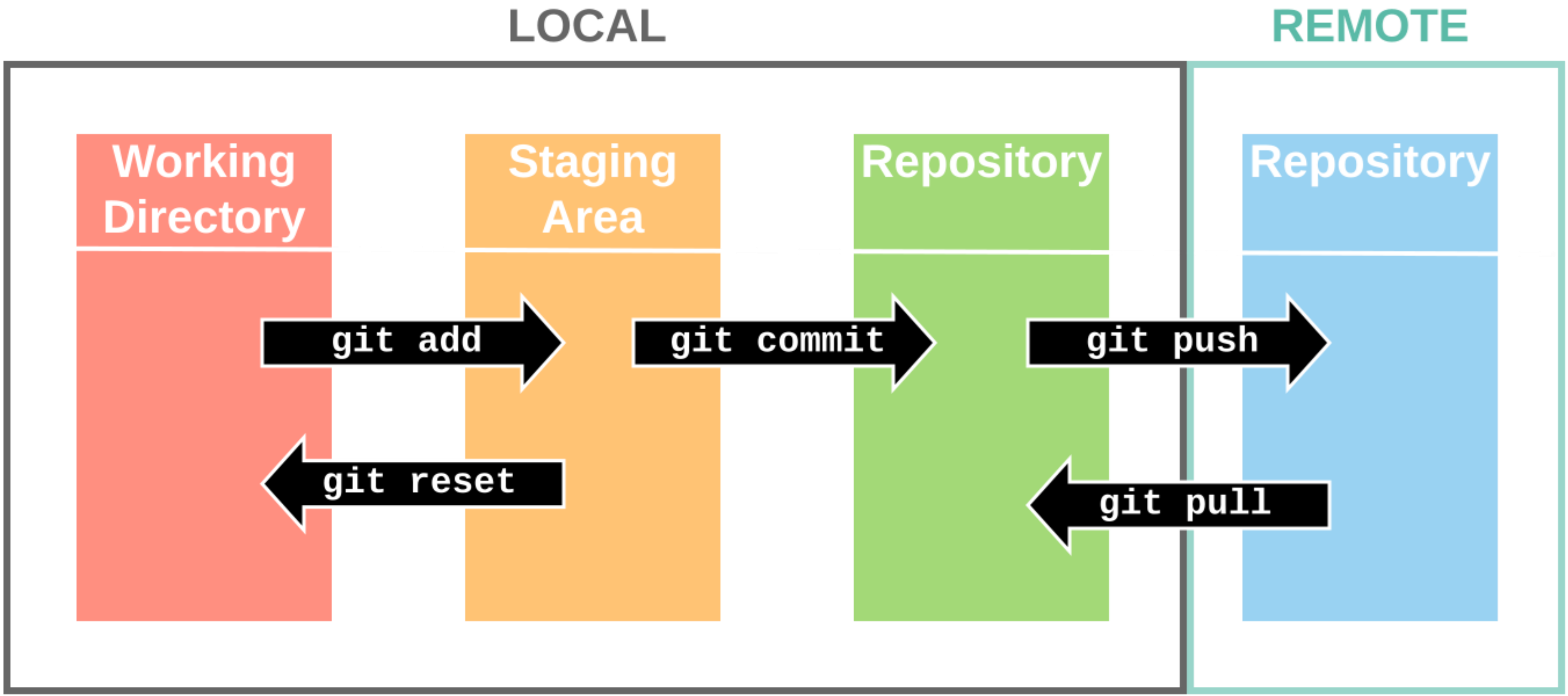
Check In Code

- git add
  - Adds a file to the staging area
  - git add -A
  - git status
  - git reset
- git commit
  - Commit the changes in the staging area to the local repository with a message
- git push
  - This action publishes your local repository to the remote repository (GitHub)
  - git push <remote> <branch>
  - Examples:
    - git push origin main
    - git push origin zach-dev

Check Out Code

- git clone
  - Copy a repository to your local machine for the first time
- git pull
  - Download remote repository
  - Update local repository to match remote repository
  - Examples:
    - git pull origin main
    - git pull origin zach-dev

Git Workflow: Check In & Check Out

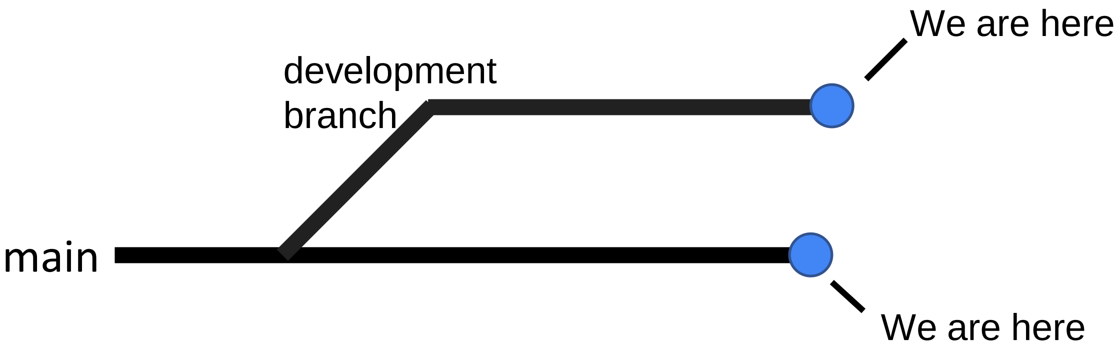


Git Workflow: Commit

- “The body of your message should provide detailed answers to the following questions: What was the motivation for the change? How does it differ from the previous implementation?” - Github FAQ

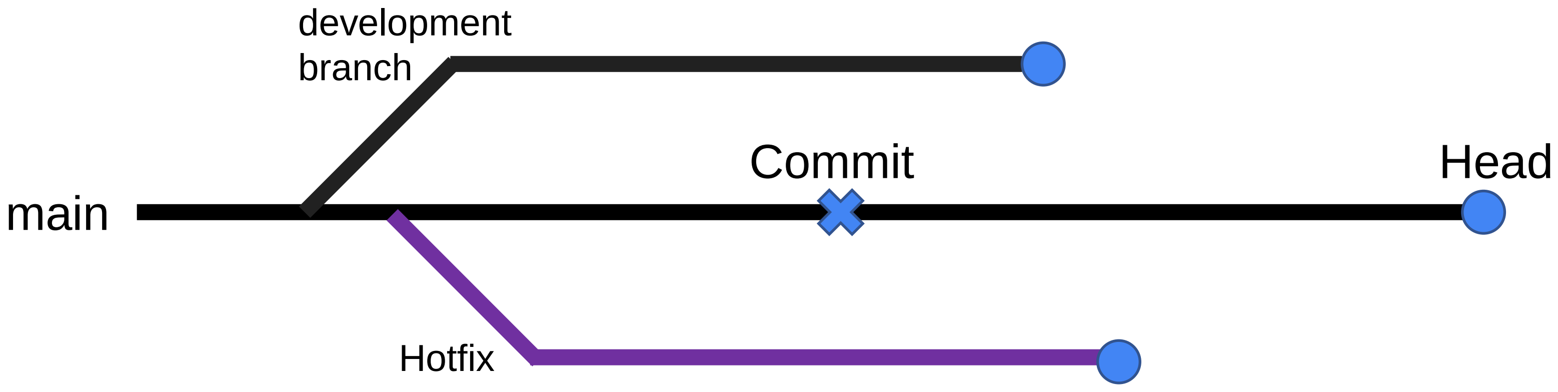
- The audience for your commit messages are developers looking to contribute to that repository
- Bad Commit : git commit –m “Some changes”
- Better Commit: git commit –m “Updated URI handlers”
- Best Commit: git commit –m “Updated URI handlers” -m “Updated URI handlers for photo searching, thumbnail generation, and deployment data streams.”

Git Workflow: Branching

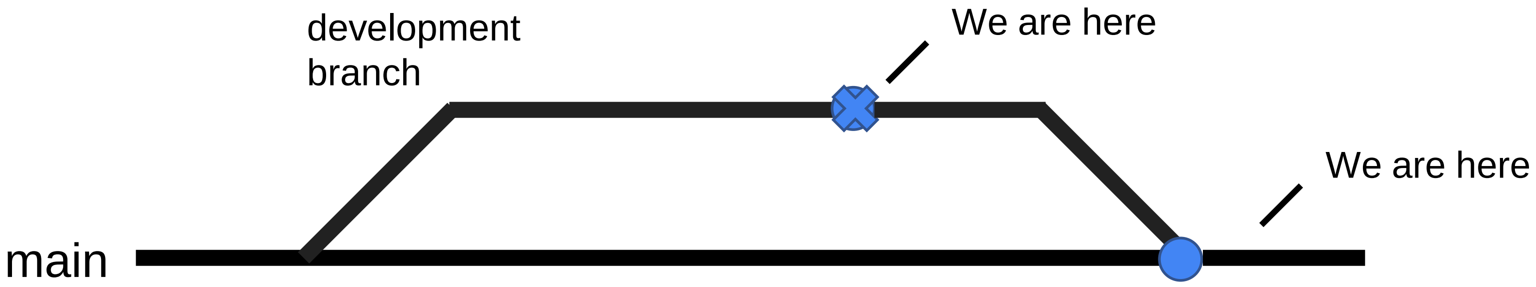


- git checkout -b “development branch”
- checkout switches the currently active branch
- -b argument creates the new branch “development branch”

Git Workflow: Branching Continued

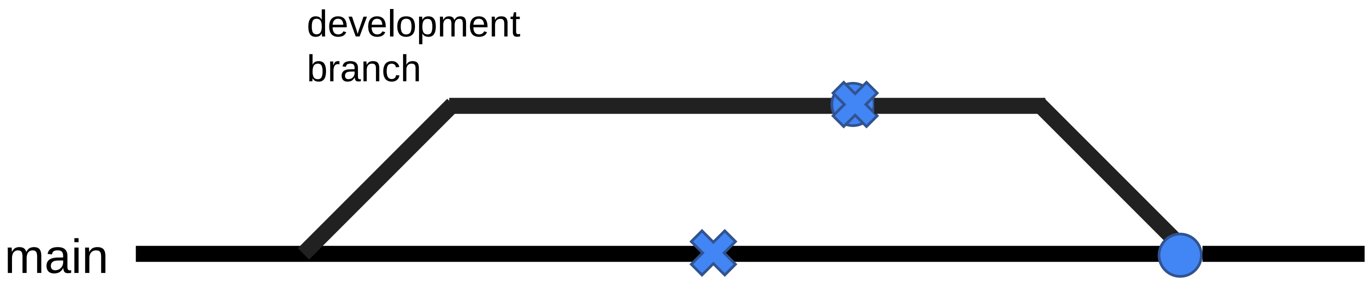


Git Workflow: Merging



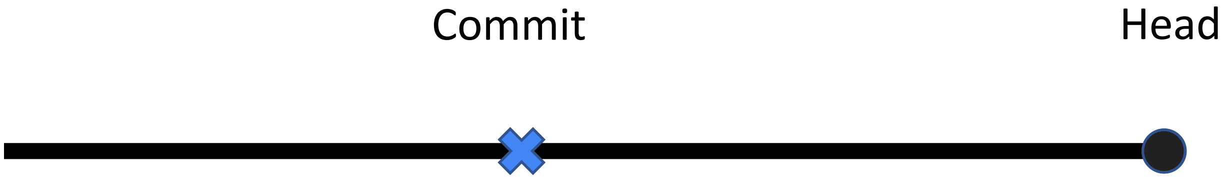
- git checkout main
  - This switches back to the main branch
- git merge “development branch”
  - This merges “development branch” into the currently active main branch
- Merges will automatically commit

Git Workflow: Handling Conflicts



- Sometimes we modify the same code in the same file
- (You have probably run into this already)
- git mergetool

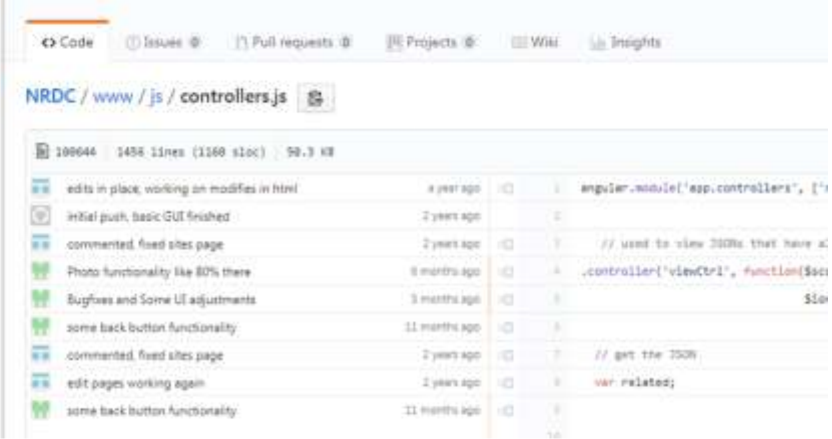
Git Workflow: Git reset --hard



- Resets the branch back to the last commit
- Dangerous on single branch
- What happens if I reset with staged changes (but uncommitted)?

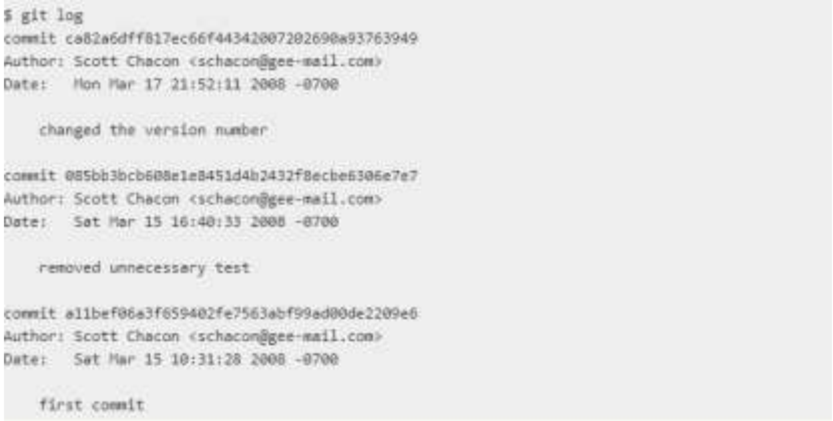
Git Workflow: Git History

- git blame
  - Who’s doing what and where?
  - We can even see this on GitHub UI?
- git log
  - Using this we can see the commit history
  - Using the commit names we can reset to a prior commit
  - git checkout <commit>



NRDC / www / js / controllers.js

```
100644 | 1456 lines (1160 sloc) | 50.3 KB
edits in place, working on modifies in html    a year ago      1  angular.module('app.controllers', ['
initial push, basic GUI finished               2 years ago     2
commented, fixed sites page                    2 years ago     3    // used to view JSONs that have a
Photo functionality like 80% there             8 months ago    4  .controller('viewCtrl', function($sc
Bugfixes and Some UI adjustments               3 months ago    5                                 $lo
some back button functionality                 11 months ago   6
commented, fixed sites page                    2 years ago     7    // get the JSON
edit pages working again                       2 years ago     8    var related;
some back button functionality                 11 months ago   9
                                                              10
```



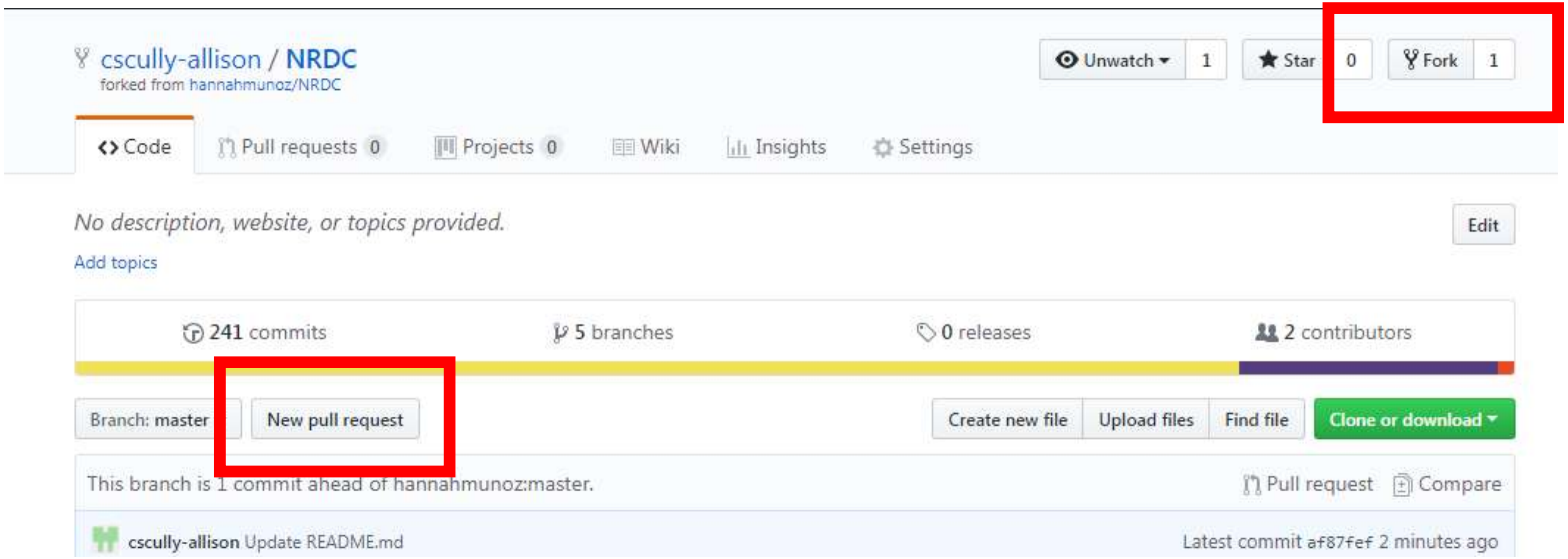
```
$ git log
commit ca82a6dff817ec66f44342007202690a93763949
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

    changed the version number

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test

commit a11bef06a3f659402fe7563abf99ad00de2209e6
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sat Mar 15 10:31:28 2008 -0700

    first commit
```

Git Tools

- GitKraken
  - GUI application
  - https://www.gitkraken.com/
- SourceTree
  - GUI application
  - https://www.sourcetreeapp.com/
- TortoiseGit
  - Integrates with Windows Explorer as right-click options
  - https://tortoisegit.org/
- Github Desktop
  - GUI Application
  - https://desktop.github.com/
- Git Large File Storage
  - Git extension for versioning large files, such as videogame art
  - https://git-lfs.github.com/

Contributing to Open Source

- Forking a Repository
- Modify and Pull Requests



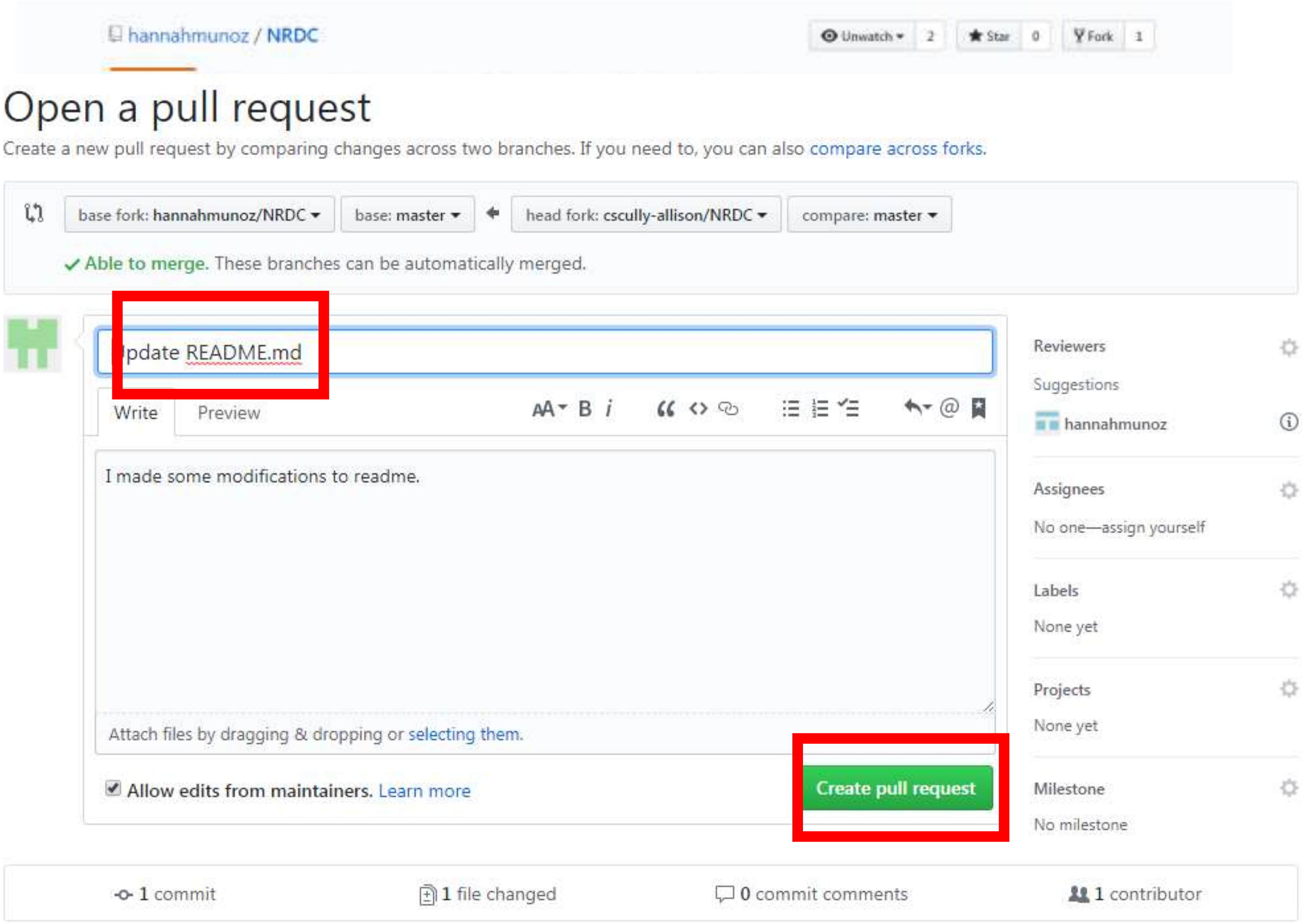
csully-allison / NRDC
forked from hannahmunoz/NRDC

No description, website, or topics provided.

This branch is 1 commit ahead of hannahmunoz:master.

csully-allison Update README.md

Latest commit af87fef 2 minutes ago



hannahmunoz / NRDC

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 fork: hannahmunoz/NRDC ▾ base: master ▾ ← head fork: cscully-allison/NRDC ▾ compare: master ▾

✓ Able to merge. These branches can be automatically merged.

Update README.md

I made some modifications to readme.

Attach files by dragging & dropping or selecting them.

☑ Allow edits from maintainers. Learn more

Reviewers

Suggestions

hannahmunoz

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

1 commit | 1 file changed | 0 commit comments | 1 contributor

Questions?