

Version control using Git

<https://github.com/nih-fmrif/git-training>

Session 3

12/19/2019

Data Science and Sharing Team
NIMH

Logistics

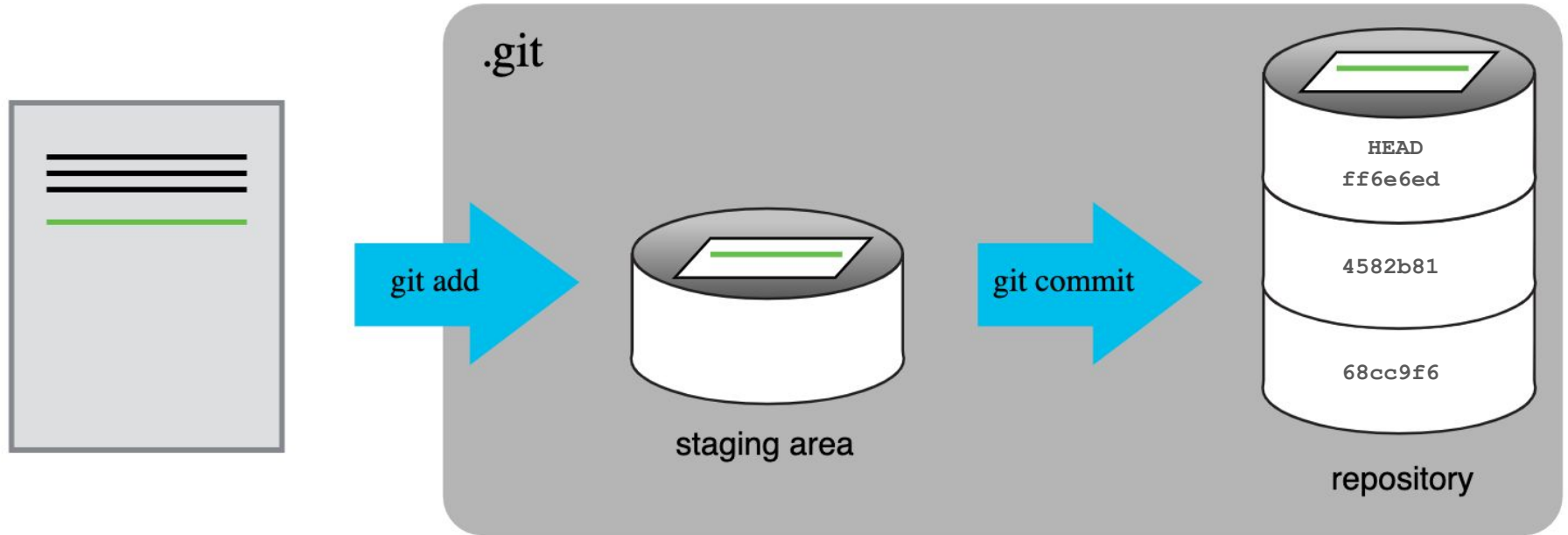
Today:

- Lecture - Review, branching, merging, pulling
- Examples: Branching and merging (non-interactive)
- Discussion - github organizations
- Exercise - merge conflicts and working on the same repo

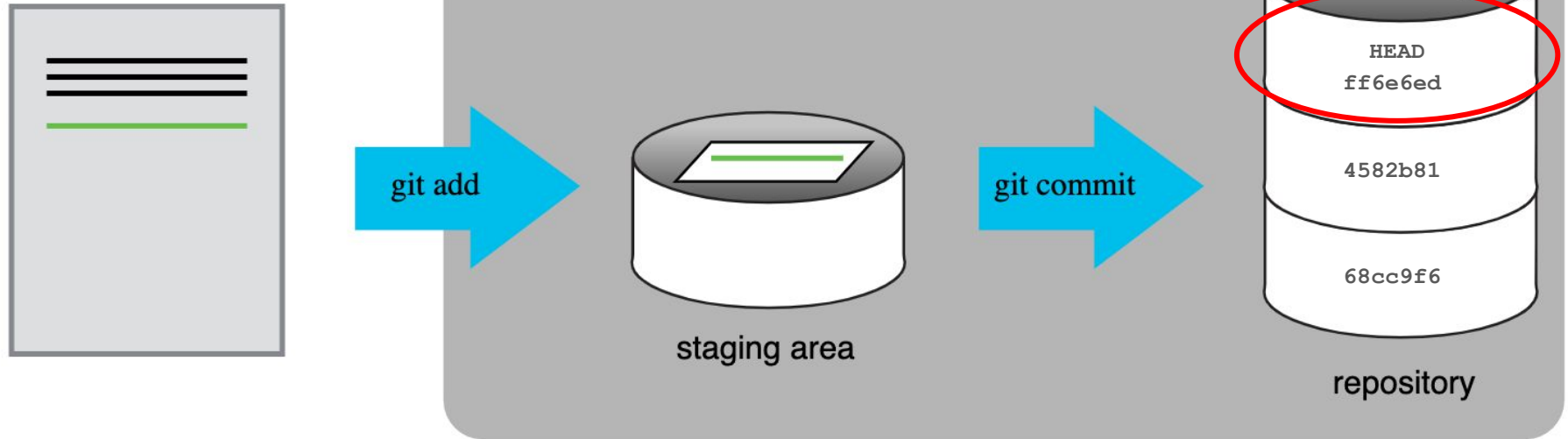
Please share your GitHub usernames: <https://tinyurl.com/tqooqmv>

<https://github.com/nih-fmrif/git-training>

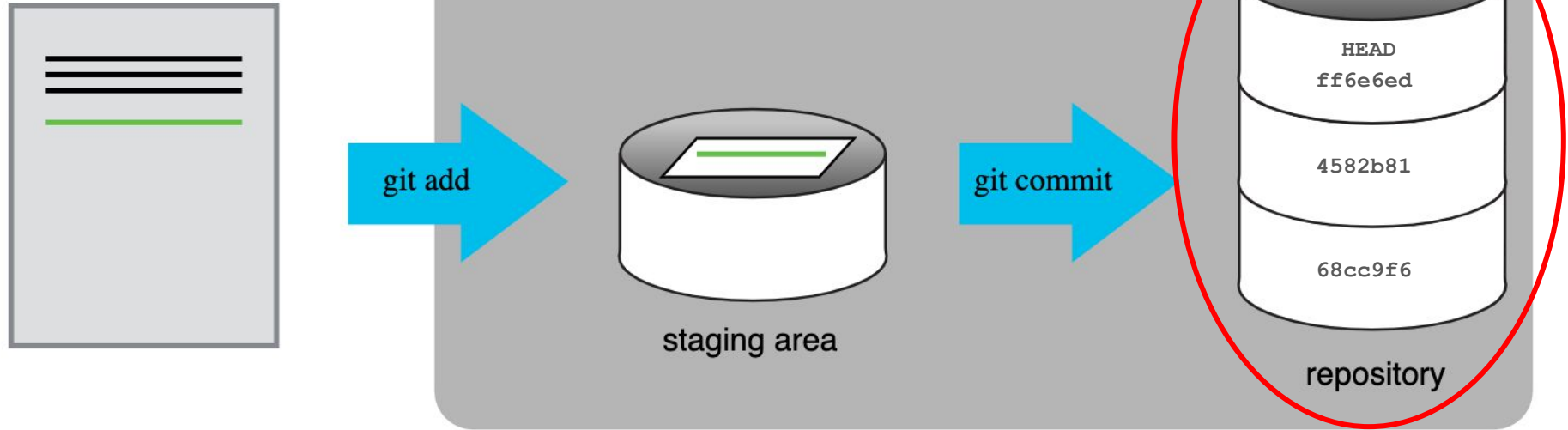
Review: repository



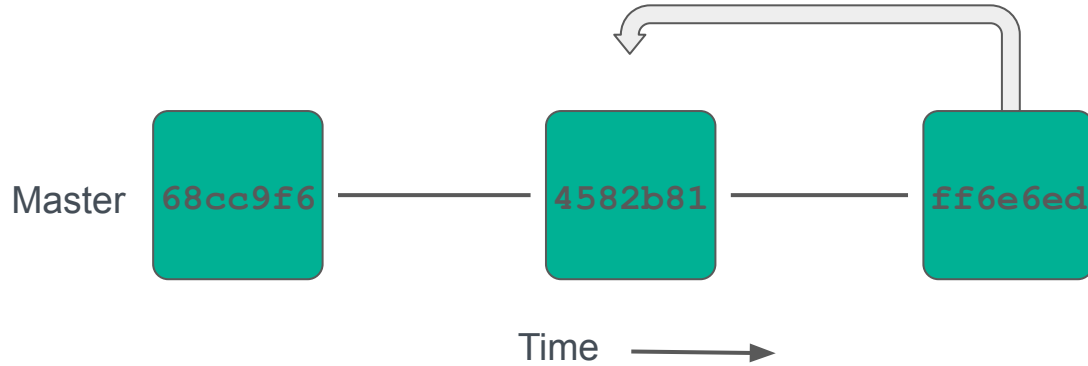
Review: repository



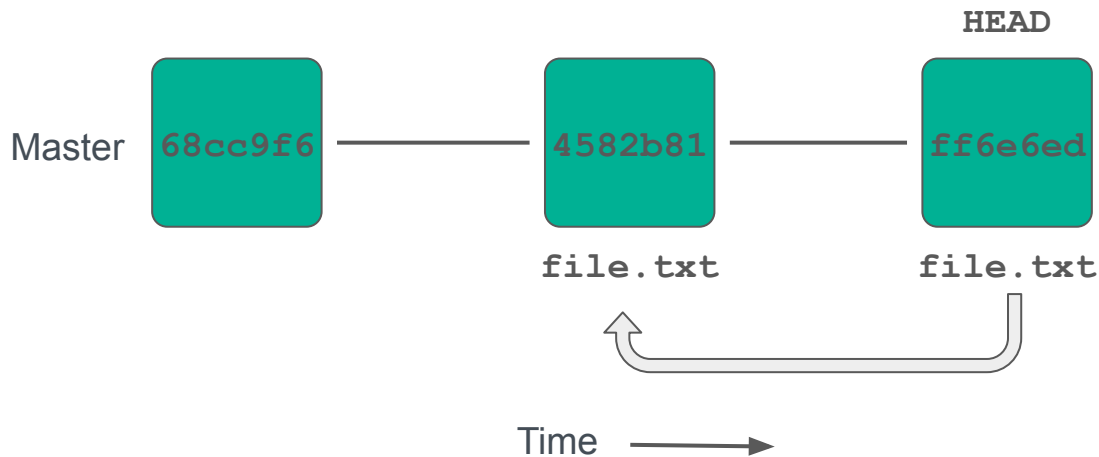
Review: repository



Review: going back in time

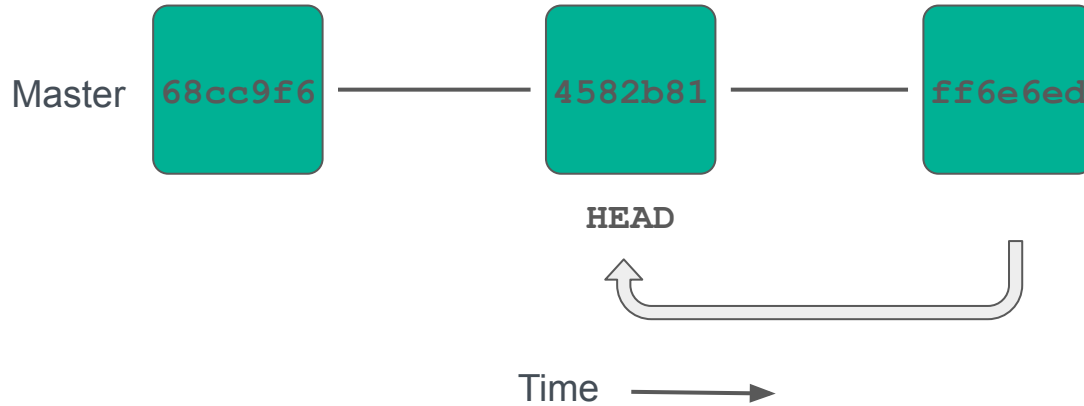


```
git checkout 4582b81 <file>
```



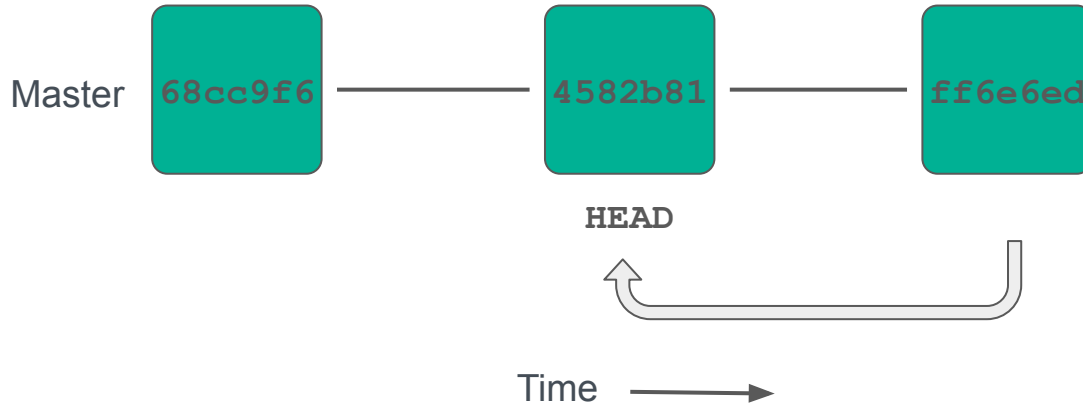
Revert `file.txt` to previous version

```
git checkout 4582b81
```



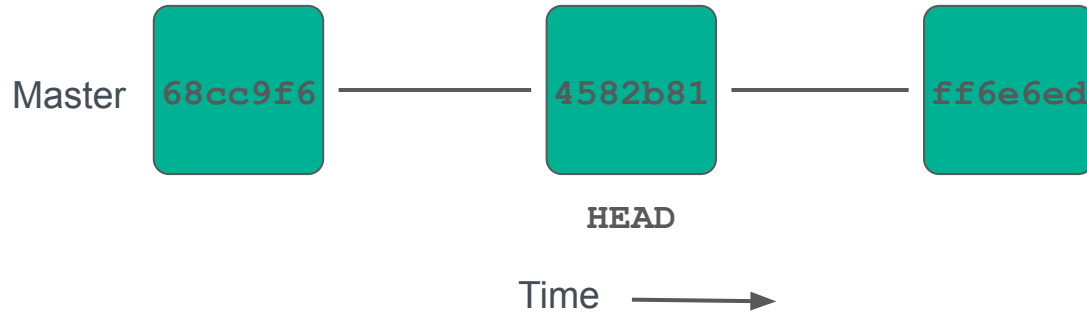
Changes HEAD to previous commit


```
git checkout 4582b81
```

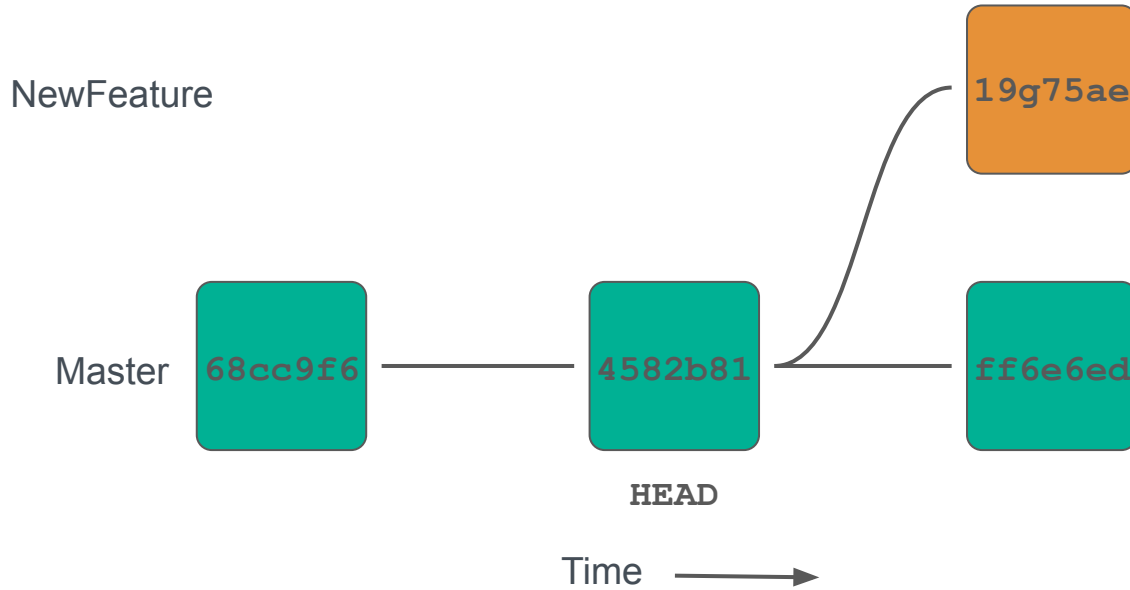


What happens if you want to change the files from a previous commit?

Git command: branch

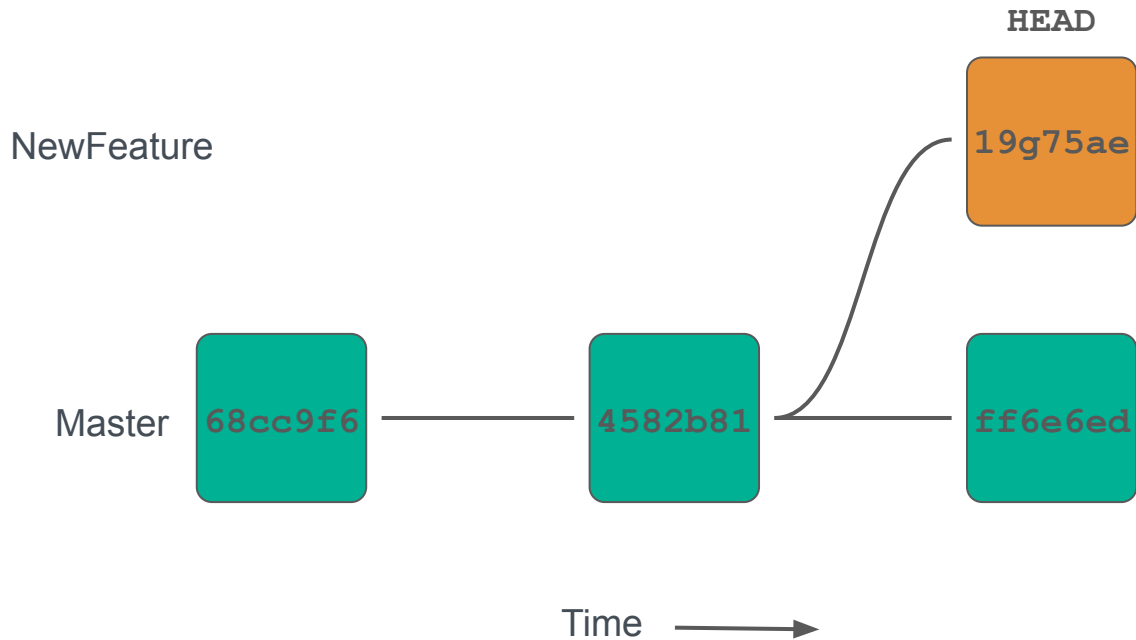


Git command: branch



```
git branch NewFeature
```

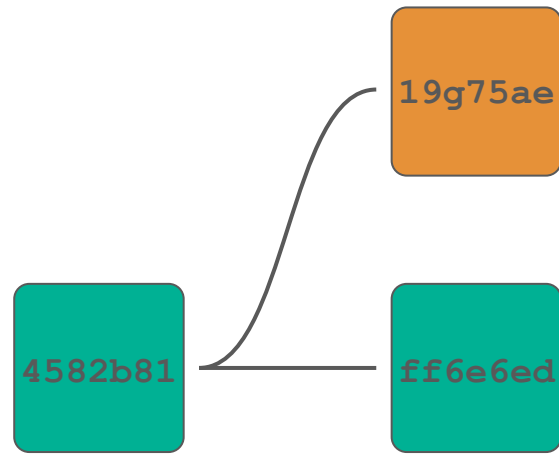
Pointing your HEAD to another branch



```
git checkout NewFeature
```

Best practices: branching

- Master branch only contains deployable code
- New branch for single item of development (eg new feature)
- Once work on branch is complete (and tested!)
 - Merge branch into master
 - Delete development branch



Example

Branching (non-interactive)

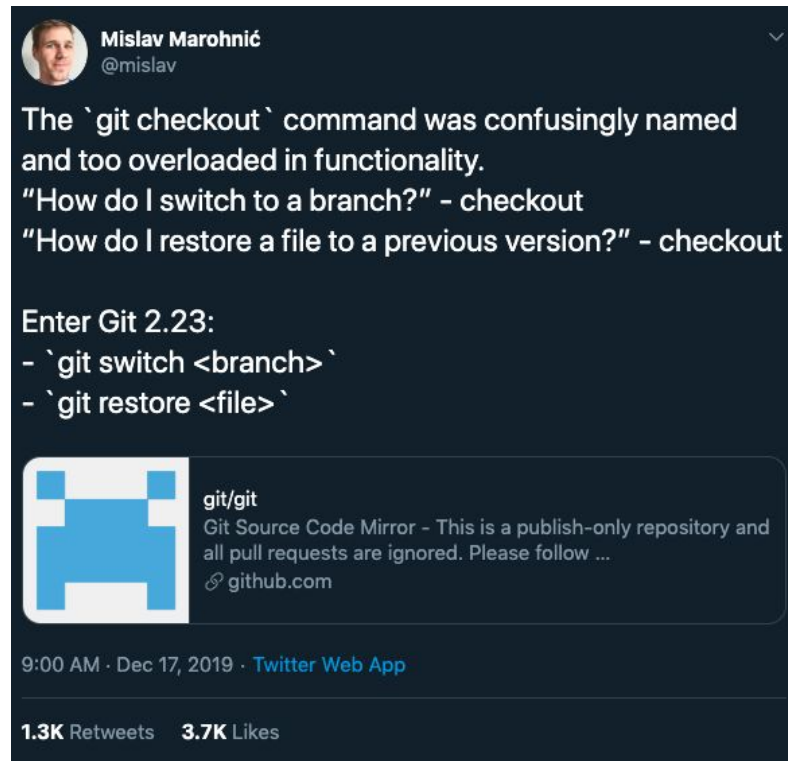
[https://github.com/nih-fmrif/git-training/tree/master/
session_3#branching-demo](https://github.com/nih-fmrif/git-training/tree/master/session_3#branching-demo)

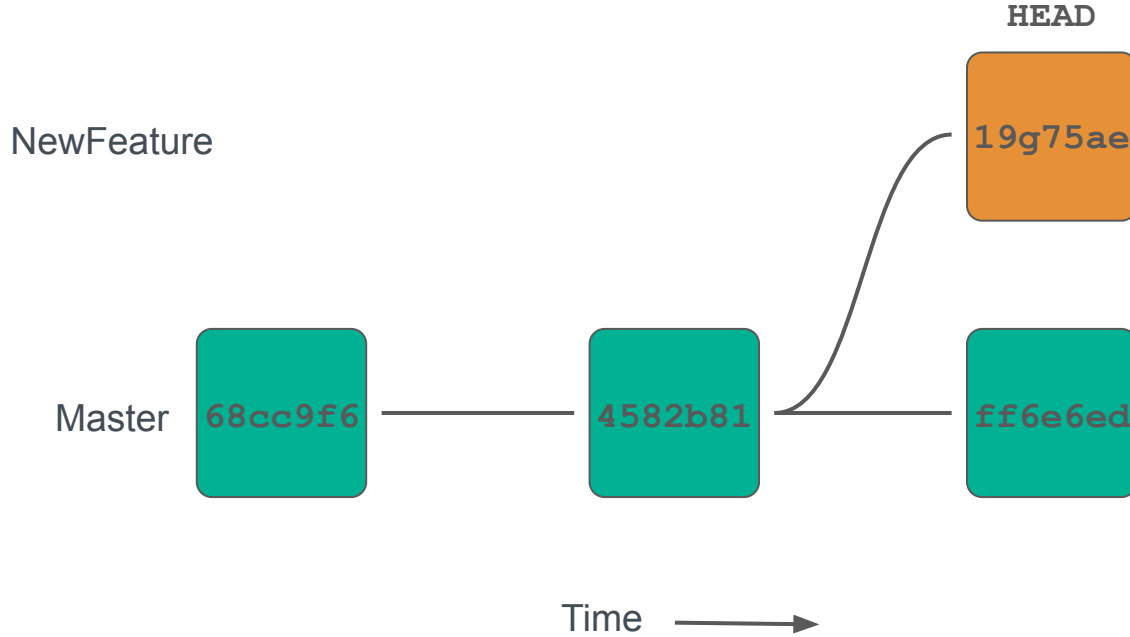
Git command: checkout

- Revert file to different version
- Inspect other commits
- Change branches

Git command: checkout

- Revert file to different version
- Inspect other commits
- Change branches
- Confusing?





How to merge two timelines into one?

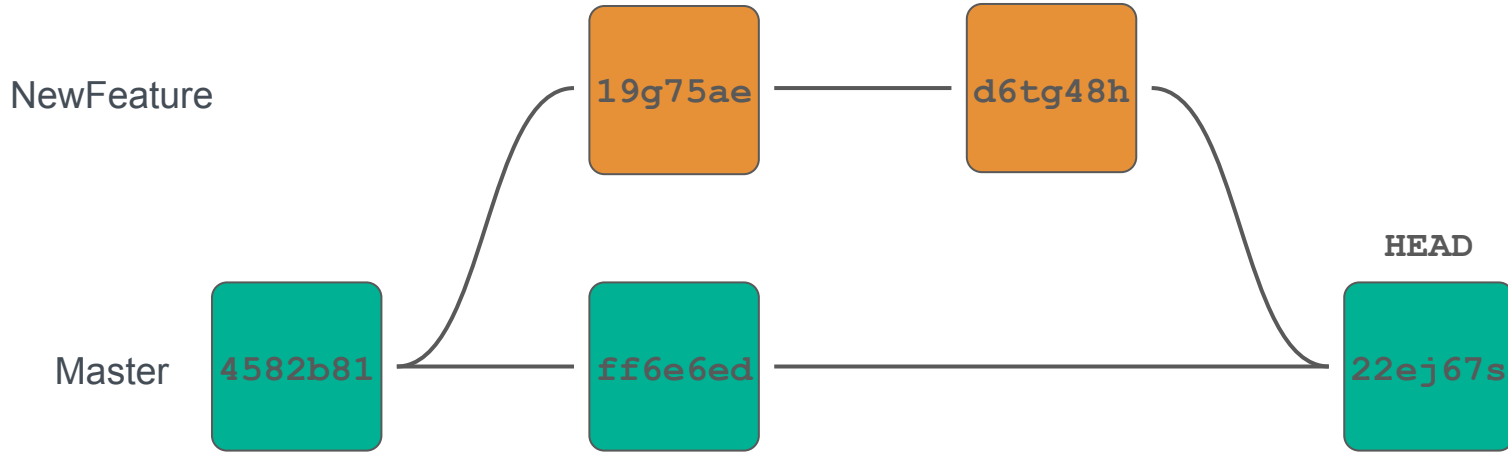
Git command: merge

Integrates independent lines of development created by `git branch` into a single branch

- Merge external branch into current branch
- Combines sequence of commits into one history
- `merge` creates a new commit

```
git merge NewFeature
```

Git command: merge

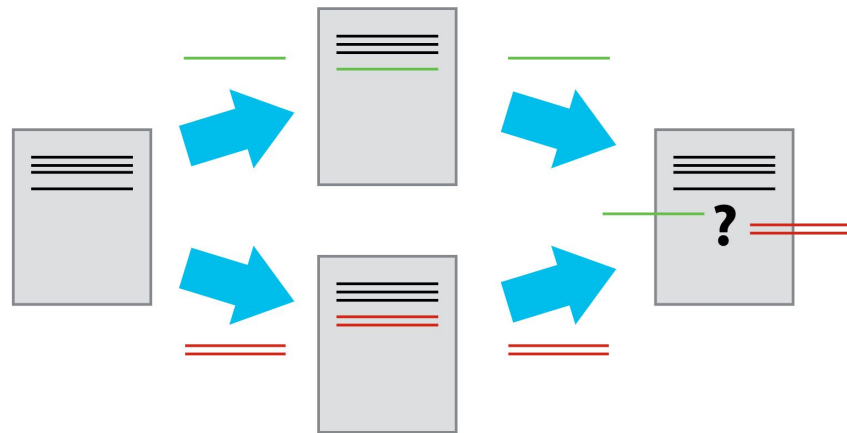


```
git checkout master  
git merge NewFeature
```

Conflicts and merging

Conflicts can occur:

- Working on your own files
 - Out of sync local and remote
 - Conflicts between branches when trying to merge
- Collaborating on the same remote



Git will try to auto resolve conflicts, however, if the *same line* differs between both changes:

- Git will add conflict markers to the conflicting file
- Decisions have to be made

Conflict markers

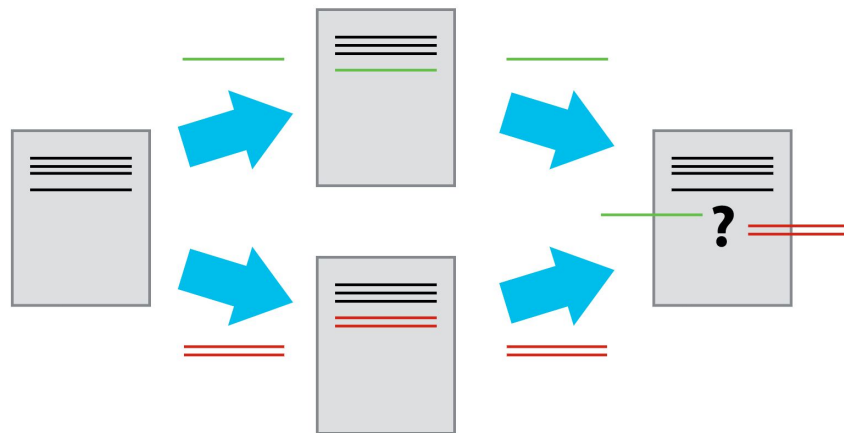
<<<<<<< HEAD

(lines in HEAD)

=====

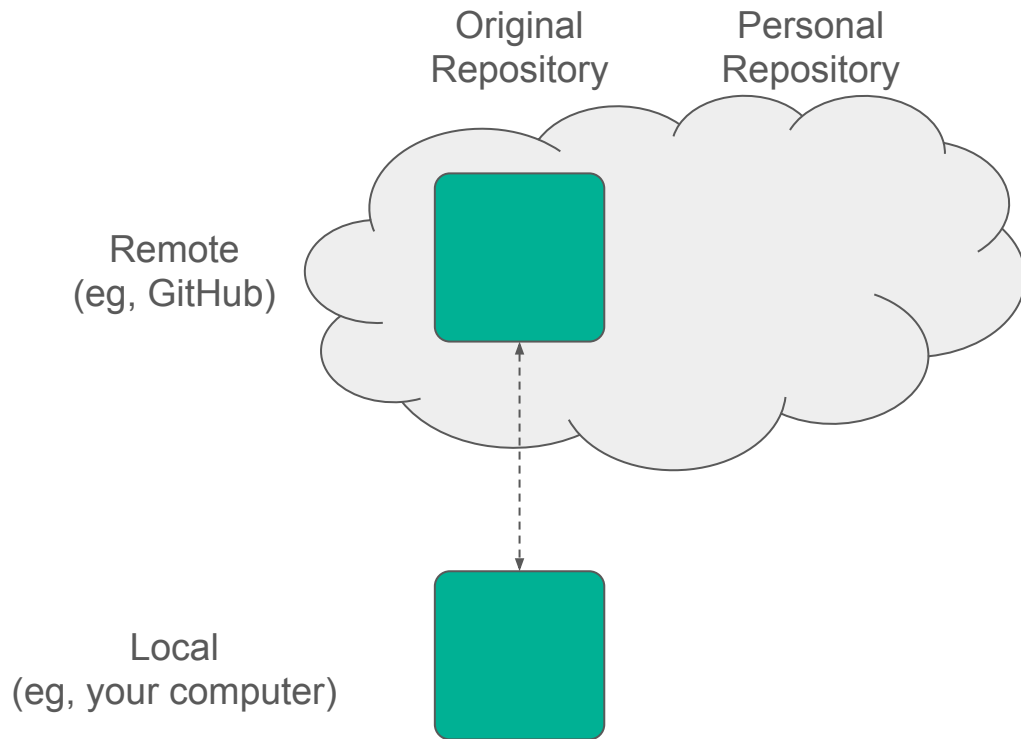
(lines in merging branch)

>>>>>>> *commitID*



The decision of which changes to keep or how to resolve the conflict must be done manually

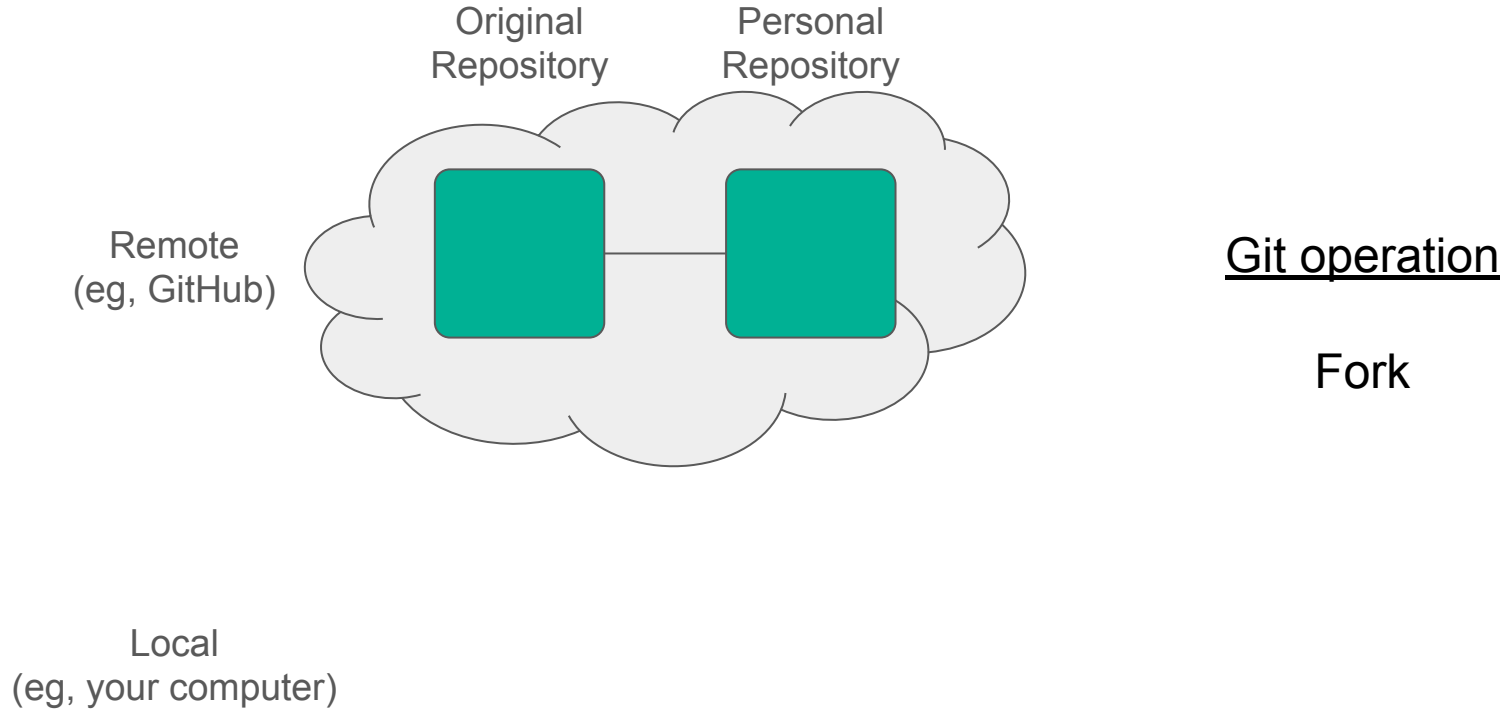
Review: Clone versus Fork



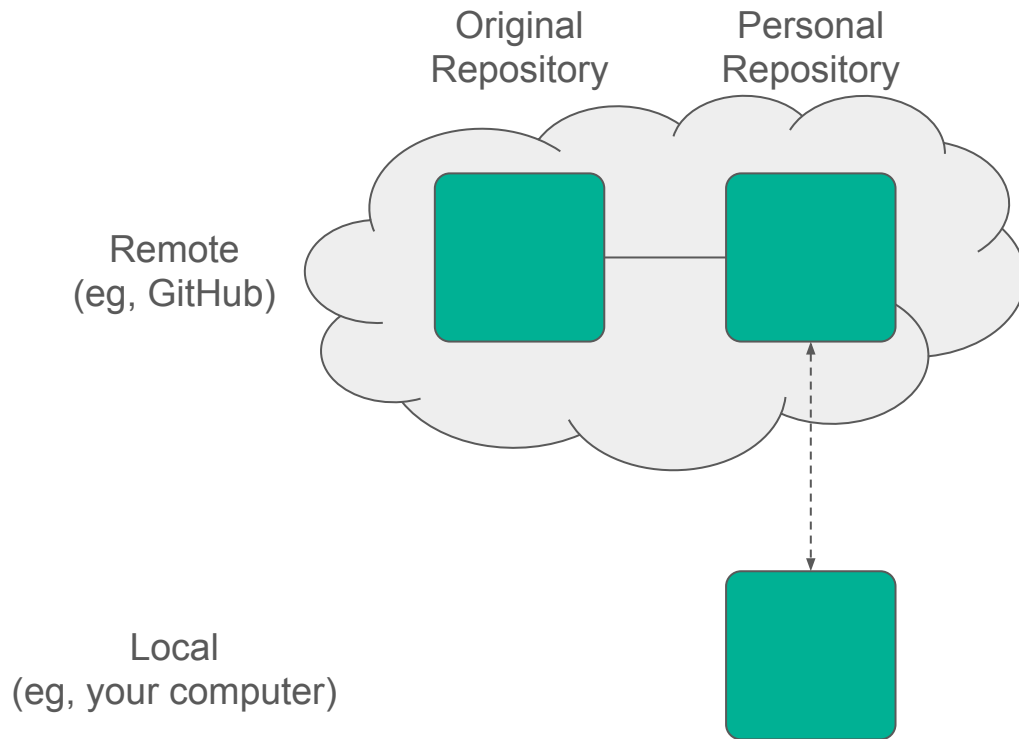
Git operation

Clone

Review: Clone versus Fork



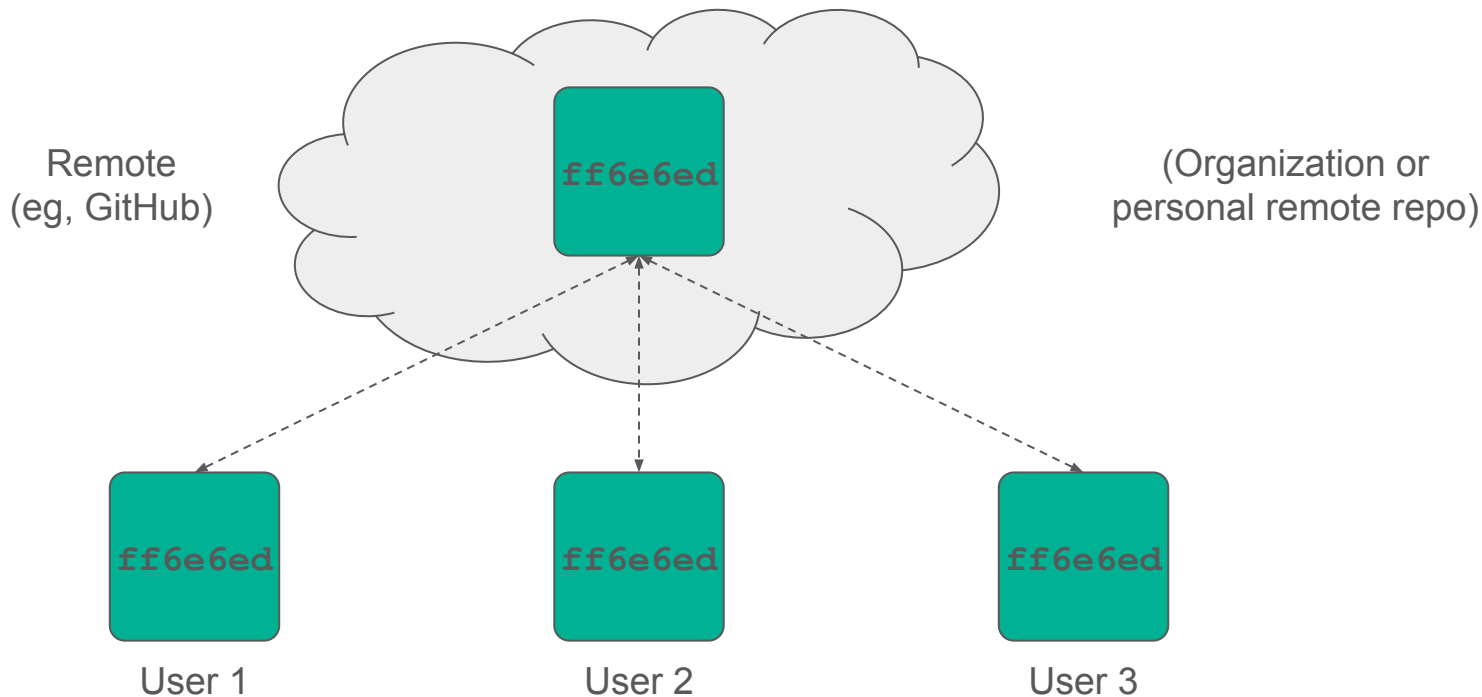
Review: Clone versus Fork



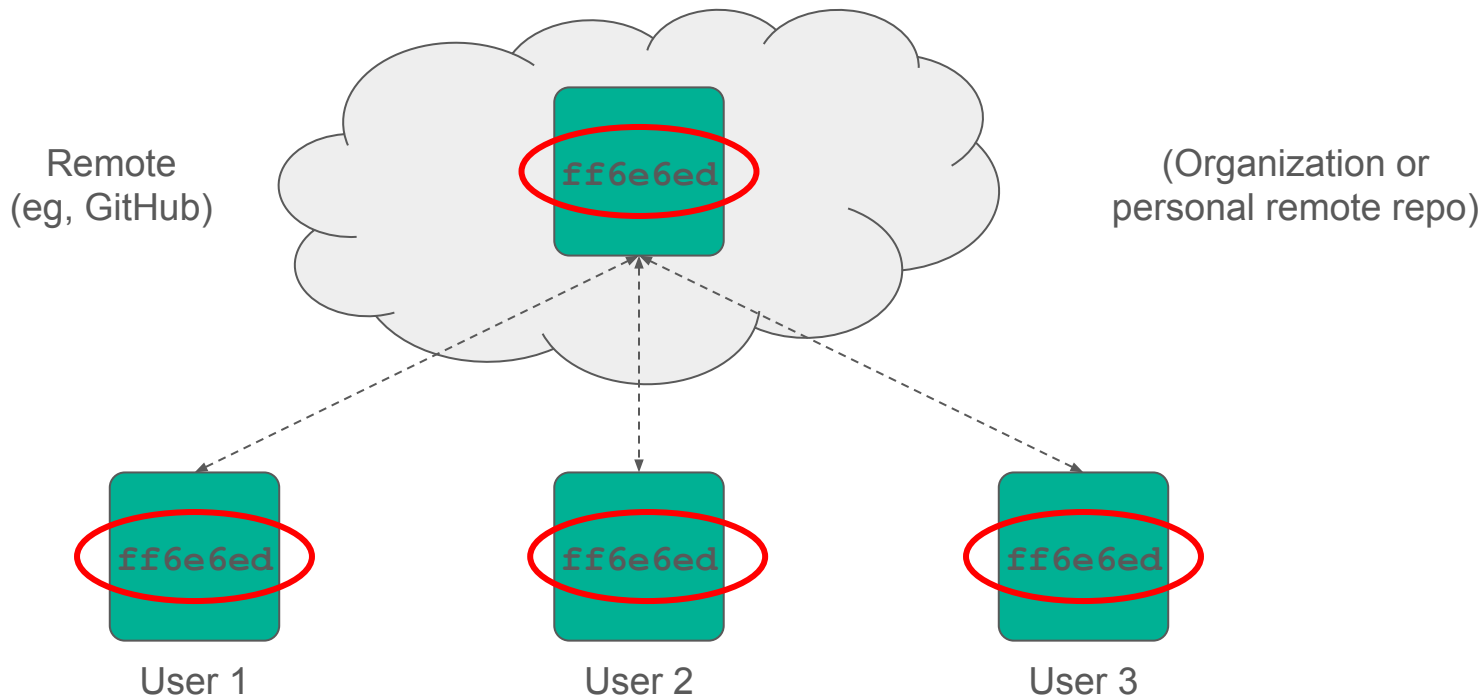
Git operation

Fork
Clone

Collaboration using same remote repo



Collaboration using same remote repo



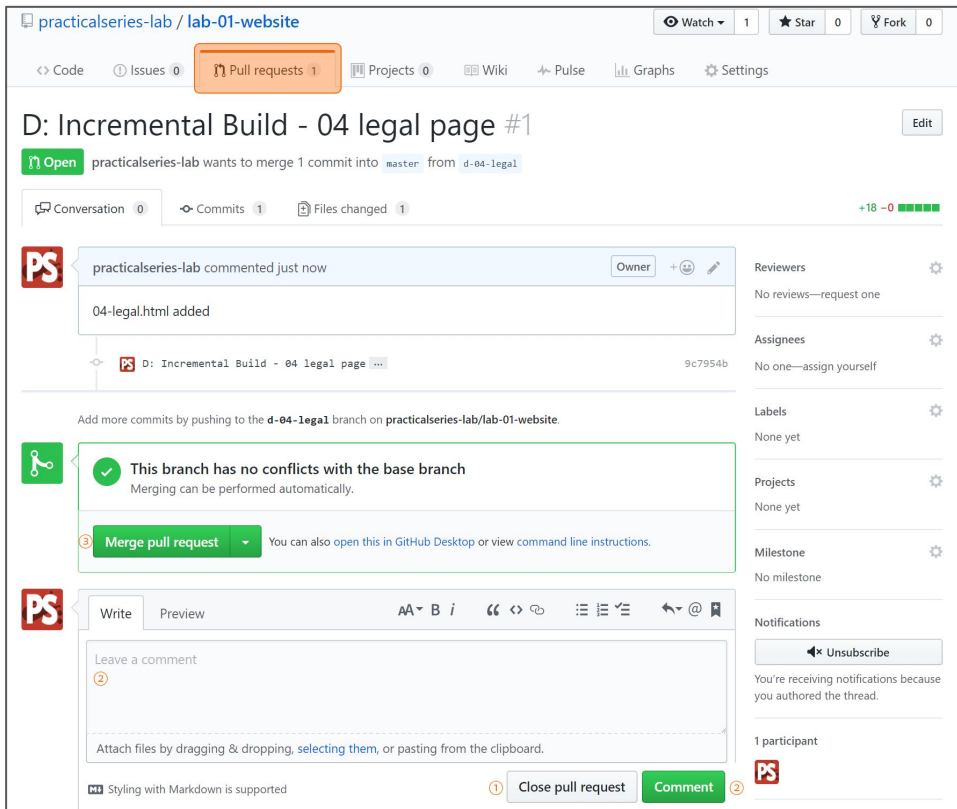
*Same commit ID

Git concept: pull request

Alerts owner of a repo that someone has made a modification and that they are *requesting* that you *pull* their modification into the repository.

- Nothing happens except the opening of the request
 - No changes to the files
- The owner has options
 - Reject
 - Discuss
 - Accept
- Pull requests can also be used to merge branches

Git concept: pull request



<http://practicalseries.com/1002-vcs/09-05-github.html#js--090503>

Example

Pull request to merge branches
(non-interactive)

[https://github.com/nih-fmrif/git-training/tree/master/
session_3#pull-request-demo](https://github.com/nih-fmrif/git-training/tree/master/session_3#pull-request-demo)

Organizations on GitHub

Choose a plan

Pick a plan for your team



Team for Open Source

\$0 USD

Collaboration tools for teams who don't need private repositories

[Choose Team for Open Source](#)

- ✓ Unlimited public repositories
- ✗ Unlimited private repositories
- ✓ GitHub Actions
See details
- ✓ GitHub Packages
See details
- ✓ Team access controls
- ✓ User management
- ✓ Issues and bug tracking
- ✓ Project management
- ✓ [Advanced tools and insights](#)



Team

\$9 USD

Per user / month

Starts at \$25 and includes 5 users

Advanced collaboration and management tools for teams

[Choose Team](#)

- ✓ Unlimited public repositories
- ✓ Unlimited private repositories
- ↑ 10,000 total Action minutes/month
See pricing details
- ↑ 2GB of GitHub Packages storage
See pricing details
- ✓ Advanced vulnerability scanning for public repositories
- ✓ Automated security updates
- ✓ GitHub Security Advisories
- ✓ Team access controls
- ✓ User management and billing
- ✓ Issues and bug tracking
- ✓ Project management
- ✓ Private GitHub Pages and Wikis
- ✓ Private protected branches
- ✓ Code owners
- ✓ Repository insights



Enterprise

\$21 USD

Per user / month

Security, compliance, and deployment controls for organizations

[Start your 14-day free trial](#)

- ← Includes everything in Team
- ↑ 50,000 total Action minutes/month
See pricing details
- ↑ 50GB of GitHub Packages storage
See pricing details
- ✓ SAML single sign-on
- ✓ Access provisioning
- ✓ Invoice billing
- ✓ 99.95% uptime SLA
- ✓ Simplified account administration
- ✓ Unified search and contributions
- ✓ Priority support
- ✓ Advanced auditing

Questions?

[Learn more about Enterprise](#)

Organizations: relevant examples

FMRIF: <https://github.com/nih-fmrif>

ThomasYeoLab: <https://github.com/ThomasYeoLab>

BIDS-standard: <https://github.com/bids-standard>

Example

Walk through collaboration and merging
(interactive)

[https://github.com/nih-fmrif/git-training/tree/master/
session_3#merge-conflicts-walkthrough](https://github.com/nih-fmrif/git-training/tree/master/session_3#merge-conflicts-walkthrough)

Supplemental: Using Git from RStudio

Tools > Global Options > Git/SVN

