Git/GitHub tutorial

Jessie Thwaites, IceCube Summer School 2024

With many slides from Aswathi Balagopal V and Jeff Weber

June 4, 2024

https://github.com/

What is version control?

Tracks and manages changes that you make to something

- *Reversibility*: the ability to back up to a previous state if you discover that some modification you did was a mistake or a bad idea.
- *Concurrency*: the ability to have many people modifying the same collection of files knowing that conflicting modifications can be detected and resolved.
- *History*: the ability to attach historical data to your data, such as explanatory comments about the intention behind each change. Even for a programmer working solo, change histories are an important aid to memory; for a multi-person project, they are a vitally important form of communication among developers.

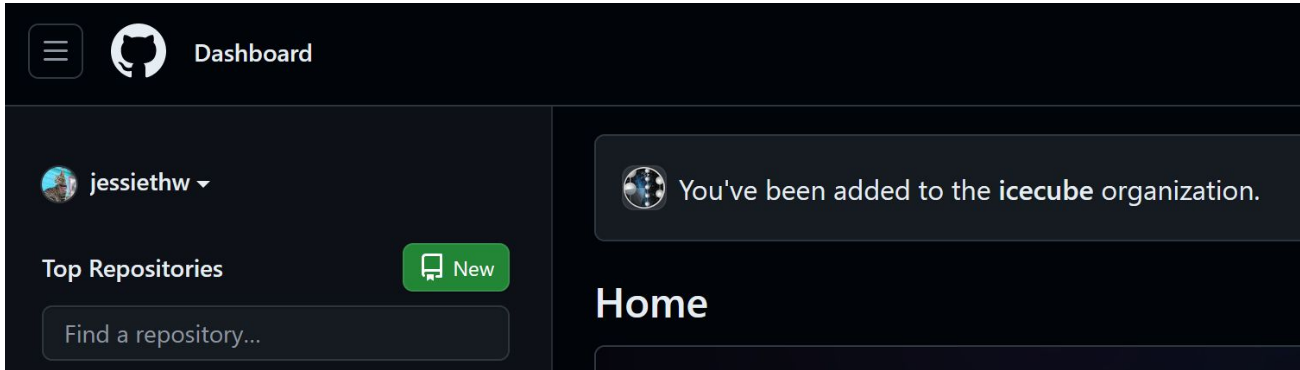
About git

- Created for linux kernel development
- Easy to use but powerful version control system
- Designed as distributed system. Manage your project on a server and work on local versions
- Keep track of different versions
- Split off different development branches and then combine them back together
- Makes collaborative work easy
- Widely used in software development

https://github.com/

About GitHub

- Web service to host remote git repositories
- Public and private repositories
- Features:
  - Forking and pull requests
  - Bugtracking, feature requests
- Used extensively in IceCube for collaboration software (IceTray, WG softwares, WG repositories)



You've been added to the **icecube** organization.

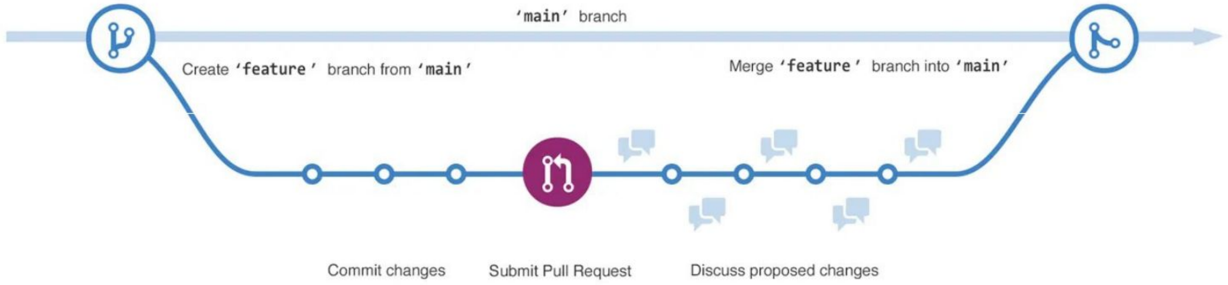
Some Terms

Repositories

- Top-level directory of files and directories that is managed by a version control system
- Often stored on a webserver
- Developers contribute to it

Branches

- Repositories can contain parallel versions of themselves
- One main branch and several development branches
- Once developed, can merge to main



Some (More) Terms

Commit

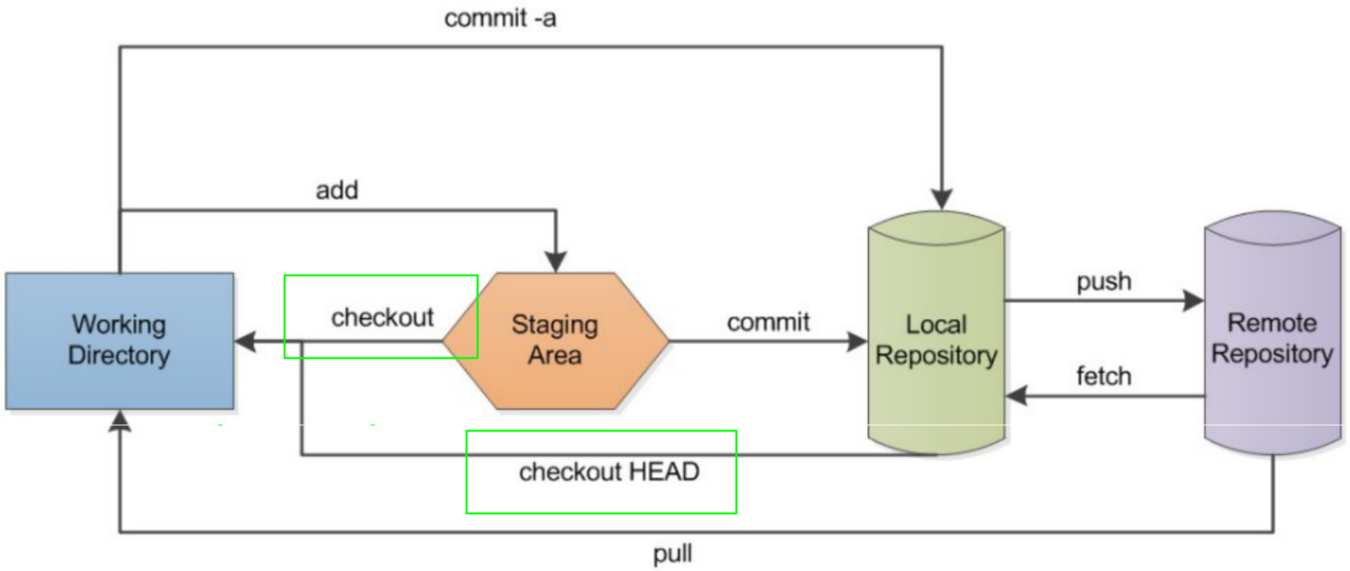
- Set of file modifications grouped under the same user-provided descriptive comment
- Provides a snapshot in time of the entire repository

Pull request

- Pull in your contribution (in your branch) and merge them into the main branch
- Better than a direct commit to the main branch to avoid mistakes

Cloning respositories

- To work with a repository you create a local copy of a remote repository
- Contains the project files and the git repository information
- Set the project files to a specific branch/version by checking it out



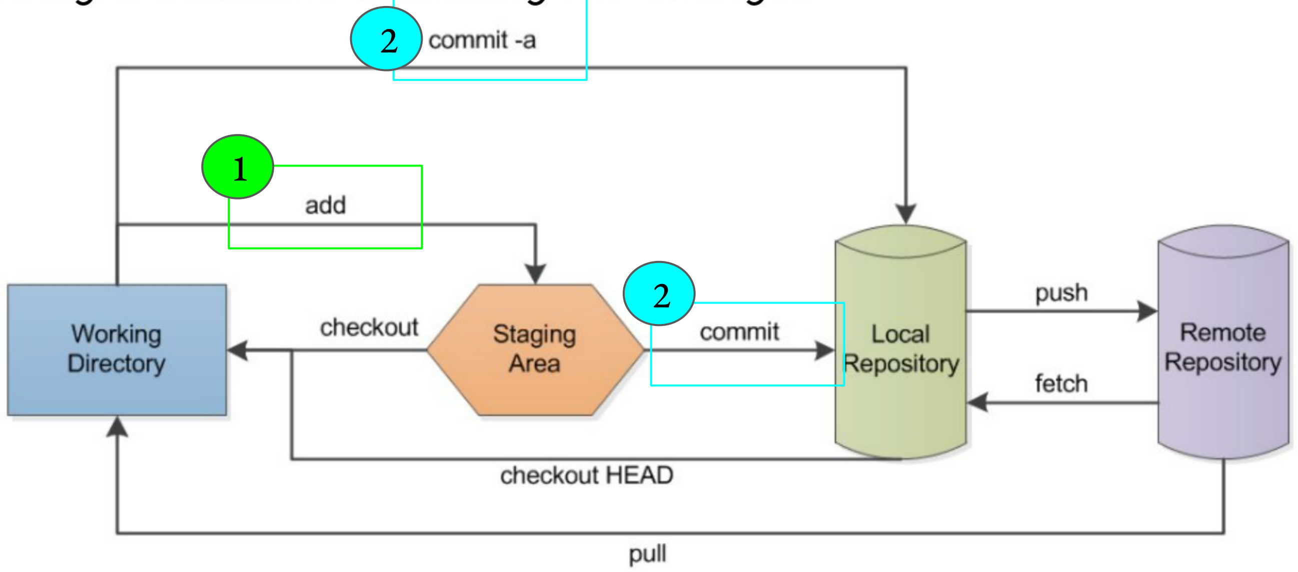
Committing changes

- If you made changes to your local files you can save them by

1 Adding them to the staging area

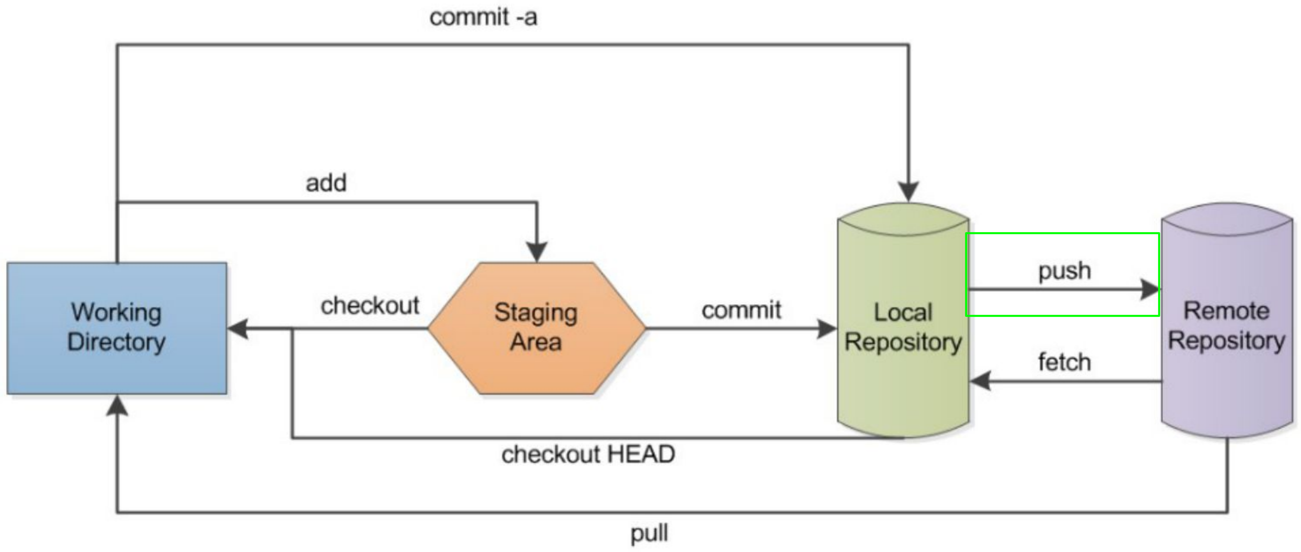
2 Committing them to your local repository

  - Writing a comment indicating the changes



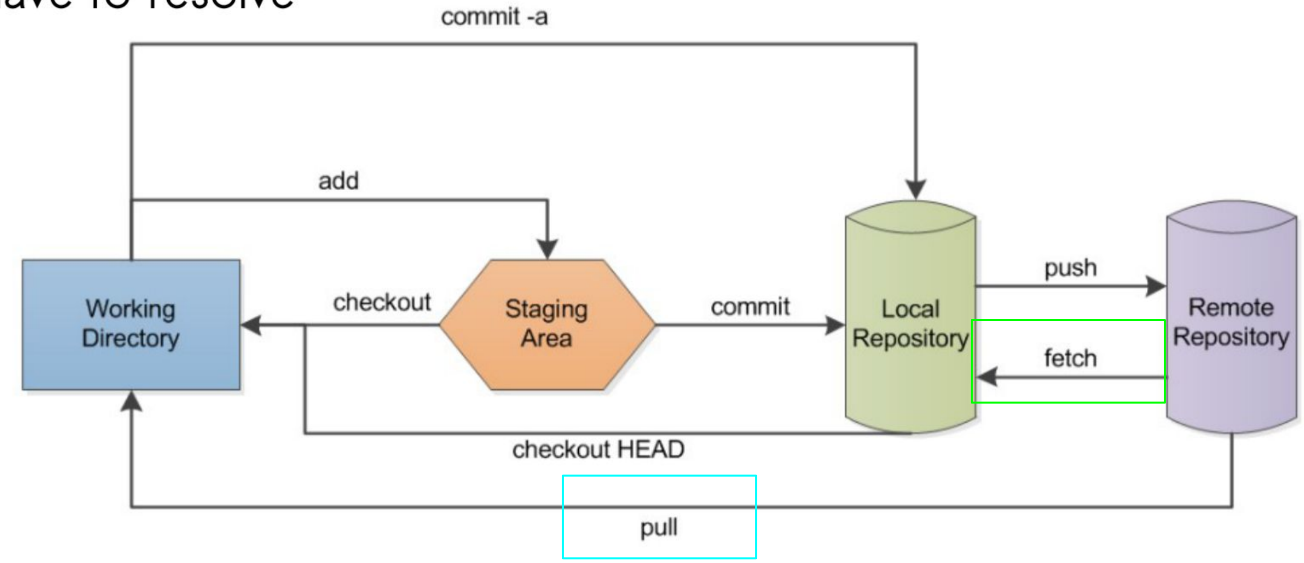
Pushing changes

- Upload your committed changes by pushing them to the upstream repository (if you have access)
- Most collaborative work use pull requests



Updating your clone

- Update your clone by fetching the latest changes from the upstream
- Update your branch by merging the fetched changes into your branch
- Or do both by invoking the pull command
- git tries to merge files automatically
- Sometimes this is not possible and you will get a conflict warning which you then have to resolve



Forks

- Fork is a new repository that shares code and visibility settings with the original “upstream” repository
- You want to contribute to a repository you do not own (e.g. some cool project)
- Create a remote copy (fork)
- Develop your fork as usual
- Send a pull request to the original repository to request merging of your changes

Summary

- Git is a powerful tool for collaborative software design.
- Many projects are hosted on GitHub, and it's super useful for code reproducibility and collaboration!
- You should now be in a position to manage own repositories as well as contribute to other ones
- There will be work time later today to start trying out GitHub/git

Helpful resources:
IceCube GitHub Guide
Slack channels: #icecube-it, #software
If you need to contact IT: help@icecube.wisc.edu

After this slide is more information/ written out tutorials for how to use GitHub/git :)

We will do these things in the afternoon session, but feel free to try it out beforehand if you want

Setup

- Create a free Github account (https://github.com/)
- To join the IceCube organization, you need to include:
  - Your full name in your GitHub account profile (J. Smith is OK)
  - Include your current institution in your account profile
- You can ask in #software or #icecube-it on Slack for an invite (we will do this as a group for Summer School!)
- Take a look at the IceCube Github Guide

Setup on your local computer

- You can install github on your computer

  Linux: `sudo apt-get install git`

  Mac: `sudo port install git`

- Also preinstalled on Cobalt (IceCube machines)
- Set your name and email for your command line client (replace stuff in quotes with your own):

  ```
  git config --global user.name "First Last"
  ```

  ```
  git config --global user.email "user@icecube.wisc.edu”
  ```

- Make sure this account is associated with your GitHub account (can have many!)

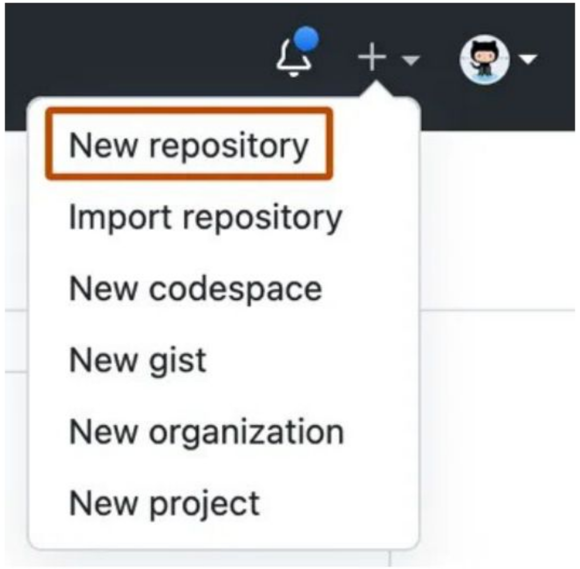
Two-factor Authentication

- Plenty of options are available. (SMS, Authenticator apps, tokens)
- GitHub will require this by end of 2023. Also requirement in IceCube
- Add and use your ssh keys
  - Nearly impossible to push commits with git on the command line otherwise
  - Follow instructions here: generating ssh keys

Using GitHub/git

Creating a Repository

- Login to github
- Click on + to create a “new repository”
- Give it a name, a description
- Choose “public/private”
- Can also add a README file, License, and .gitignore



Cloning

Go to the local directory where you want the clone and do `git clone <url>`

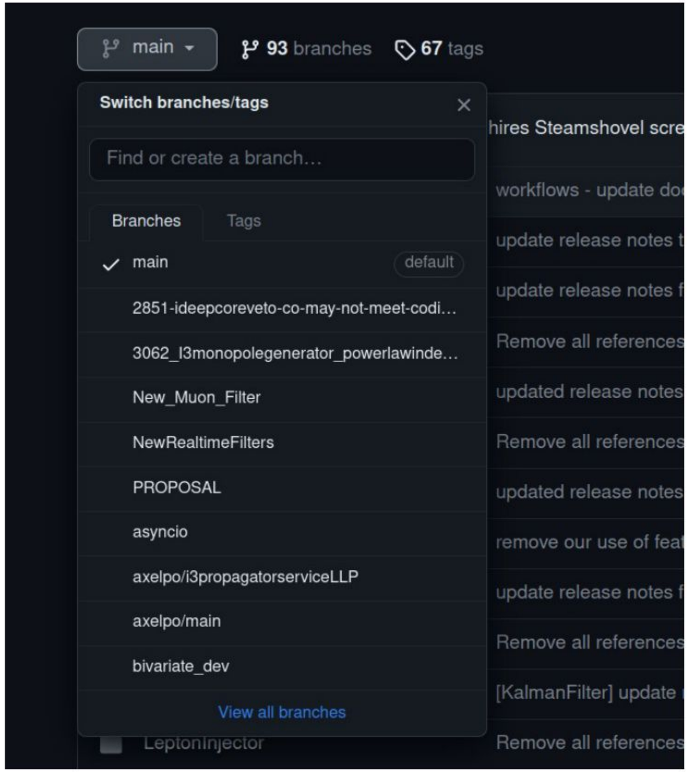
The url can be found from the repository page on github

Try cloning a repo!

- IceTray: https://github.com/icecube/icetray
- Csky (for neutrino sources): https://github.com/icecube/csky
- Some examples: https://github.com/jessiethw/summer_school_examples (from previous Summer Schools)

Working with branches

- A new repository starts with a master branch
- If you clone a repository you also start out in the master branch
- You can view branches in github interface or local with
  - git branch – a



main

93 branches

67 tags

Switch branches/tags

Find or create a branch…

Branches

Tags

✓ main default

2851-ideepcoreveto-co-may-not-meet-codi…

3062_l3monopolegenerator_powerlawinde…

New_Muon_Filter

NewRealtimeFilters

PROPOSAL

asyncio

axelpo/i3propagatorserviceLLP

axelpo/main

bivariate_dev

View all branches

LeptonInjector

hires Steamshovel scre

workflows - update do

update release notes t

update release notes f

Remove all references

updated release notes

Remove all references

updated release notes

remove our use of feat

update release notes f

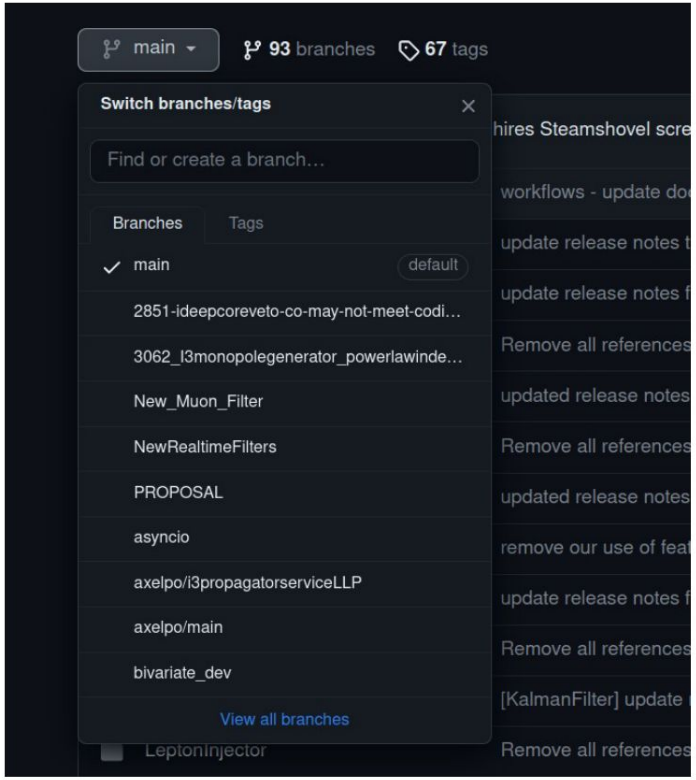
Remove all references

[KalmanFilter] update

Remove all references

Creating branches

- You can create new branches via github interface
- Using the dropdown menu switch to the branch from which you want to start
- To update your local repo do
  - git fetch



Branches at the command line

You can also create new branches via the command line

```
    git branch <branchname>

    git checkout <branchname>
```

FYI: these two commands can be combined with

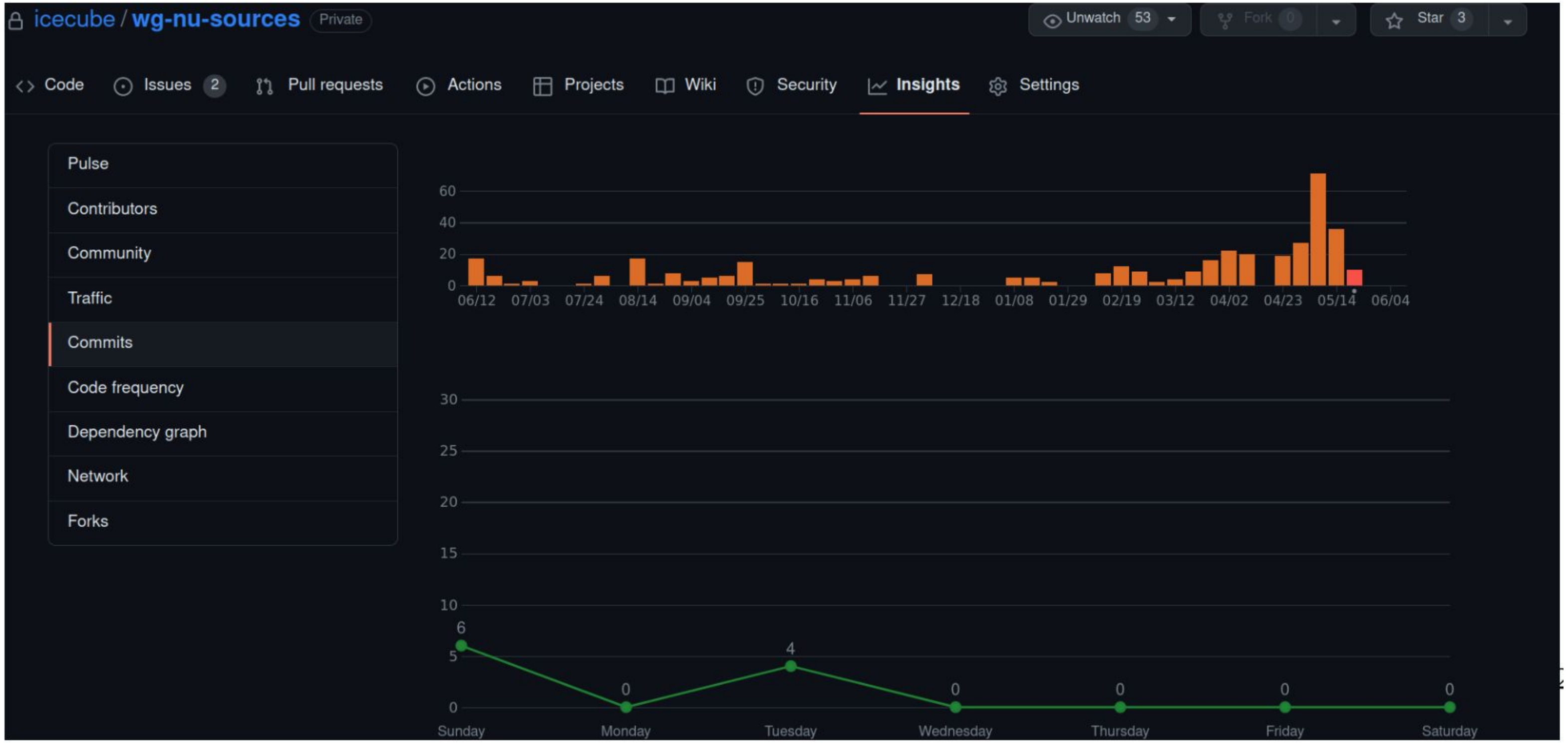
```
    git checkout -b <branchname>
```

You can switch between branches in your staging area with

```
    git checkout <branchname>
```

Exploring branches

- You can look at the branches and how they are connected using the github interface



Commits

- Committing with the github interface is not the usual case
- Usually you work on your computer and want to commit the changes to the remote repository
- To do this first switch to a branch you want to work on
- Do all your developing
- Today you could edit the Readme file and create another new file
- To see the changes of local files with respect to the last commit you can do:

```
git status
```

- It will list new and changed files

How to stage and commit

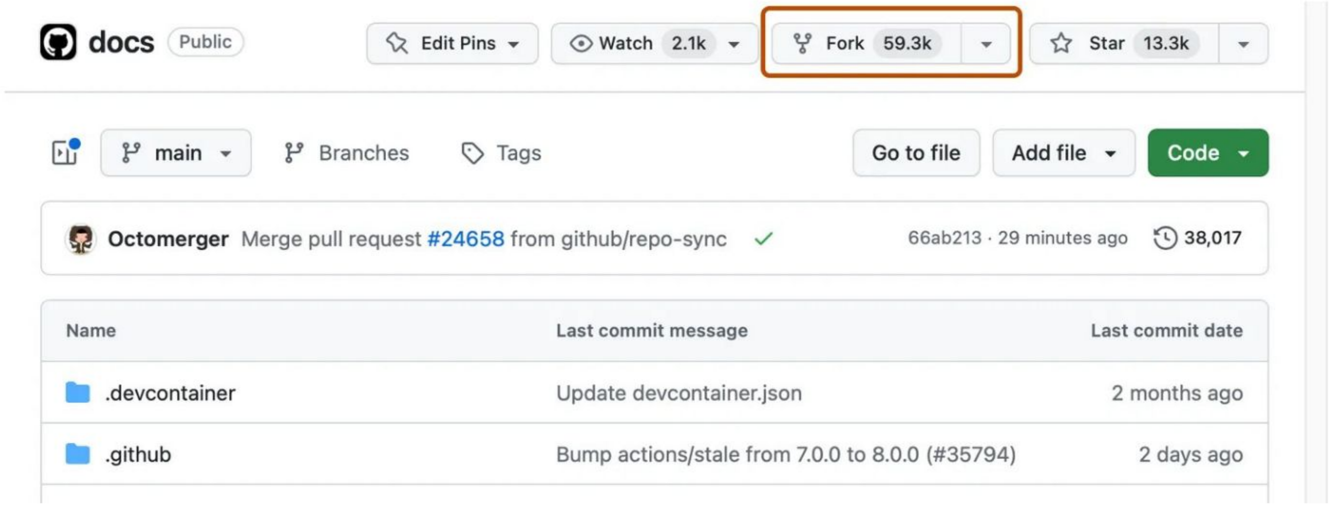
- Tell git which files to commit to the repository (stage changes): `git add <filename>`
- To commit the changes to your local repository: `git commit -m “some comment”`
- Upload the files to the remote: `git push`
- Often the remote changes between your last pull and the push. So you can `git pull` before pushing
- You can push to a branch on upstream too: `git push --set-upstream origin <branchname>`
- You can also remove added files from this list (before committing): `git reset HEAD <filename>`

Merge conflicts

- Sometimes git can not merge files because of conflicting changes
- We can simulate this
  - create a new branch from your current one, but do not switch branches
  - edit the Readme file in the current branch and commit the changes
  - change to the new branch
  - edit the Readme in the same line (with some other edit) and also commit the changes
  - Try to merge the first branch into the second
- You will get an error indicating conflicts in the file
- Edit the file to resolve the conflict (conflicting lines are indicated by «« and
  - »»)
- Add the resolved file to the staging area and commit the resolution

Forking

If you want to fork a repository in github, you can use the fork button & then clone your fork and work as usual



Octomerger Merge pull request #24658 from github/repo-sync ✓ 66ab213 · 29 minutes ago 38,017

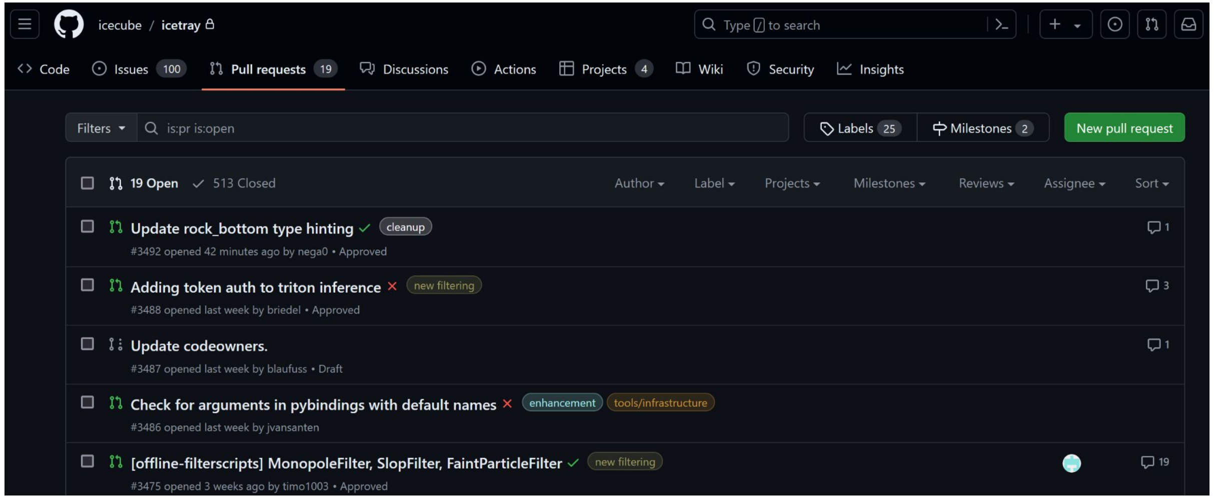
Name	Last commit message	Last commit date
.devcontainer	Update devcontainer.json	2 months ago
.github	Bump actions/stale from 7.0.0 to 8.0.0 (#35794)	2 days ago

Syncing your fork

- To keep your fork up-to-date with respect to the original you have to set the original as upstream: `git remote -v`
- Add the original as additional upstream:
  ```
  git remote add upstream <originalurl>
  git remote -v
  ```
- To fetch the original updates do: `git fetch upstream`
- Merge the original branch in your upstream branch:
  ```
  git merge upstream/<branchname>
  ```
- All changes are committed to your fork

Pull Requests (PRs)

- To merge your branch with main or your fork back with the original you have to send a pull request via the "New pull request" button. Here you should describe your changes
- The owner of the original will see this, can discuss the changes with you and ultimately accept your request.



is:pr is:open

19 Open ✓ 513 Closed

Update rock_bottom type hinting ✓ cleanup — 1
#3492 opened 42 minutes ago by nega0 • Approved

Adding token auth to triton inference ✗ new filtering — 3
#3488 opened last week by briedel • Approved

Update codeowners. — 1
#3487 opened last week by blaufuss • Draft

Check for arguments in pybindings with default names ✗ enhancement tools/infrastructure
#3486 opened last week by jvansanten

[offline-filterscripts] MonopoleFilter, SlopFilter, FaintParticleFilter ✓ new filtering — 19
#3475 opened 3 weeks ago by timo1003 • Approved

Pull request (demo)

- Checkout the summer school repository (https://github.com/jessiethw/summer_school_examples)
- Create a branch
- Add a file in the branch or make some changes
- Commit changes in the branch to upstream
- Push to your branch
- Create a pull request to merge with the main