Lecture 2: Strengthening our Skills

-

GitHub, Python, Jetstream

Instructor: Dylan Chapp

THE UNIVERSITY OF
TENNESSEE
KNOXVILLE

Today's Topics and Problems

- Learn git and GitHub
- Build your private GitHub repository
  - Define the structure of your GitHub repository
  - Push your solution of Problem 1a in your GitHub repository
- Push the solution of assignment 1 to your private GitHub
  - If you have not yet, push the complete the solution of Assignment 1 to your private GitHub repository
- Strengthen your expertise in Python and problem solving with Assignment 2
  - Clone Assignment 2 from the course GitHub repository
  - Work on the three math problems in Assignment 2
  - Implement the solutions in Python with the Jupyter Notebook
  - Push your solution to your private GitHub repository

DUE DATE: Monday 17th before 8AM ET

Git, GitHub, and (sequential) Text Manipulation

Today we will explore:

- Learn git basics
  - Creating a git repo
  - Adding files
  - Pushing changes
  - Branching
  - Syncing & Merging
  - Undoing
- Learn GitHub basics
  - Pull Requests
- Strengthen your problem solving skills:
  - Cool math problems with different solutions, each with different performance

Git Basics

Material built from:

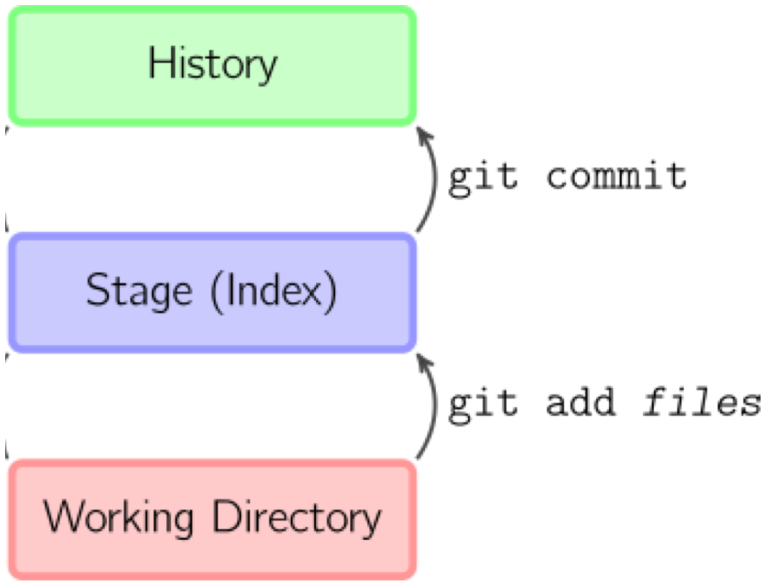
- http://rogerdudler.github.io/git-guide/
- https://marklodato.github.io/visual-git-guide/index-en.html

Creating a Git Repository

- If you are starting from scratch
  - git init
- If you already have a repo on a remote server
  - git clone username@host:/path/to/repository

Adding Files

- You can propose changes (add it to the **Index**) using:
  - git add <filename>
  - git add .
- You can commit these changes using:
  - git commit -m "Commit message”

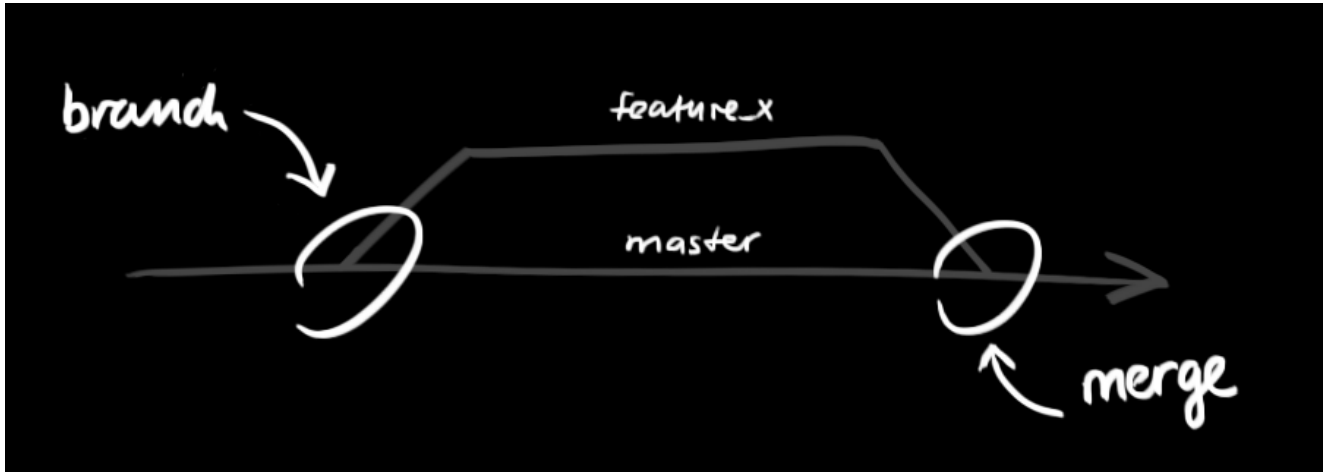


Pushing Your Changes

- Your changes are now in the **HEAD** of your local working copy
- To send those changes to your remote repository:
  - git push origin master
  - git push <remote> <branch>
- If you used ‘git init’, first specify a remote server:
  - git remote add origin <server>

Branching

- Branches are used to develop features isolated from each other.
- The *master* branch is the "default" branch when you create a repository.
- Use other branches for development and merge them back to the master branch upon completion.



Branching

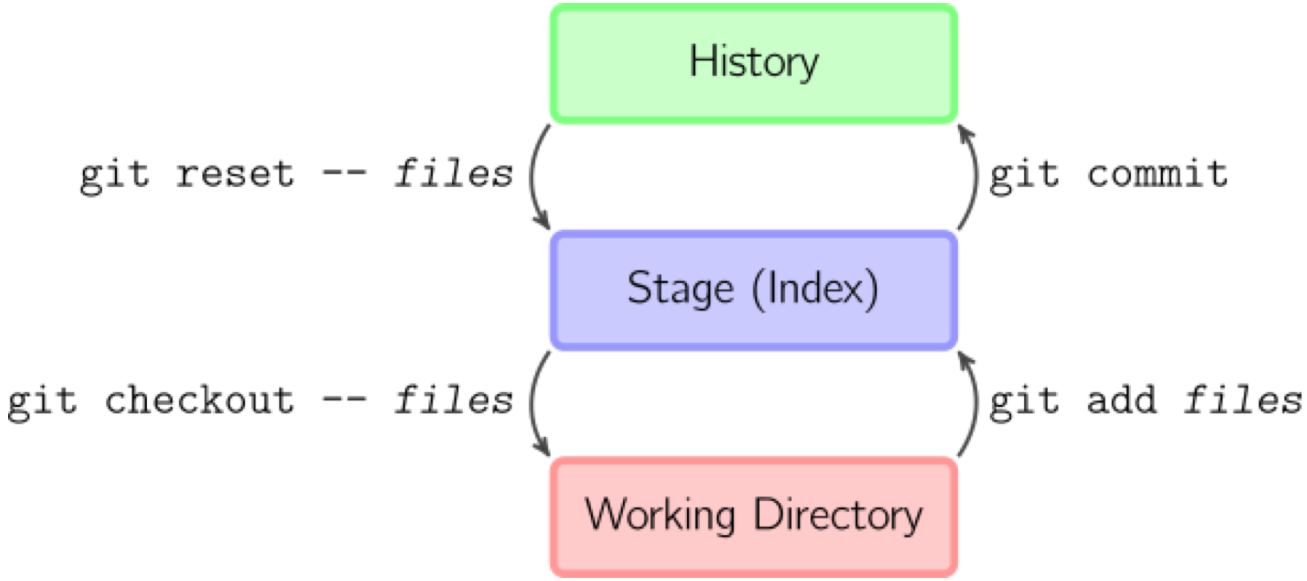
- Create a new branch named "feature_x" and switch to it:
  - git checkout -b feature_x
- Switching back to master:
  - git checkout master
- Delete the feature branch:
  - git branch -d feature_x
- Pushing a branch for others to use:
  - git push origin <branch>

Syncing & Merging

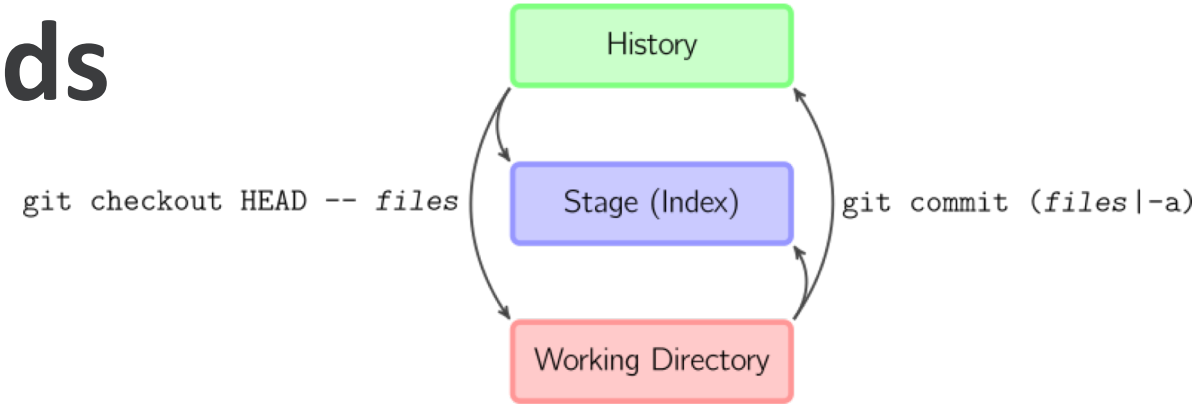
- Updating your local repository to the newest commit:
  - git pull
- Merge another branch into your active branch (e.g. master):
  - git merge <branch>
- After resolving merge conflicts, you need to mark them as merged:
  - git add <filename>
- You can also preview them by using:
  - git diff <source_branch> <target_branch>

Undoing

- Unstage changes:
  - git reset HEAD <file>...
- Discard changes in working directory:
  - git checkout -- <file> …

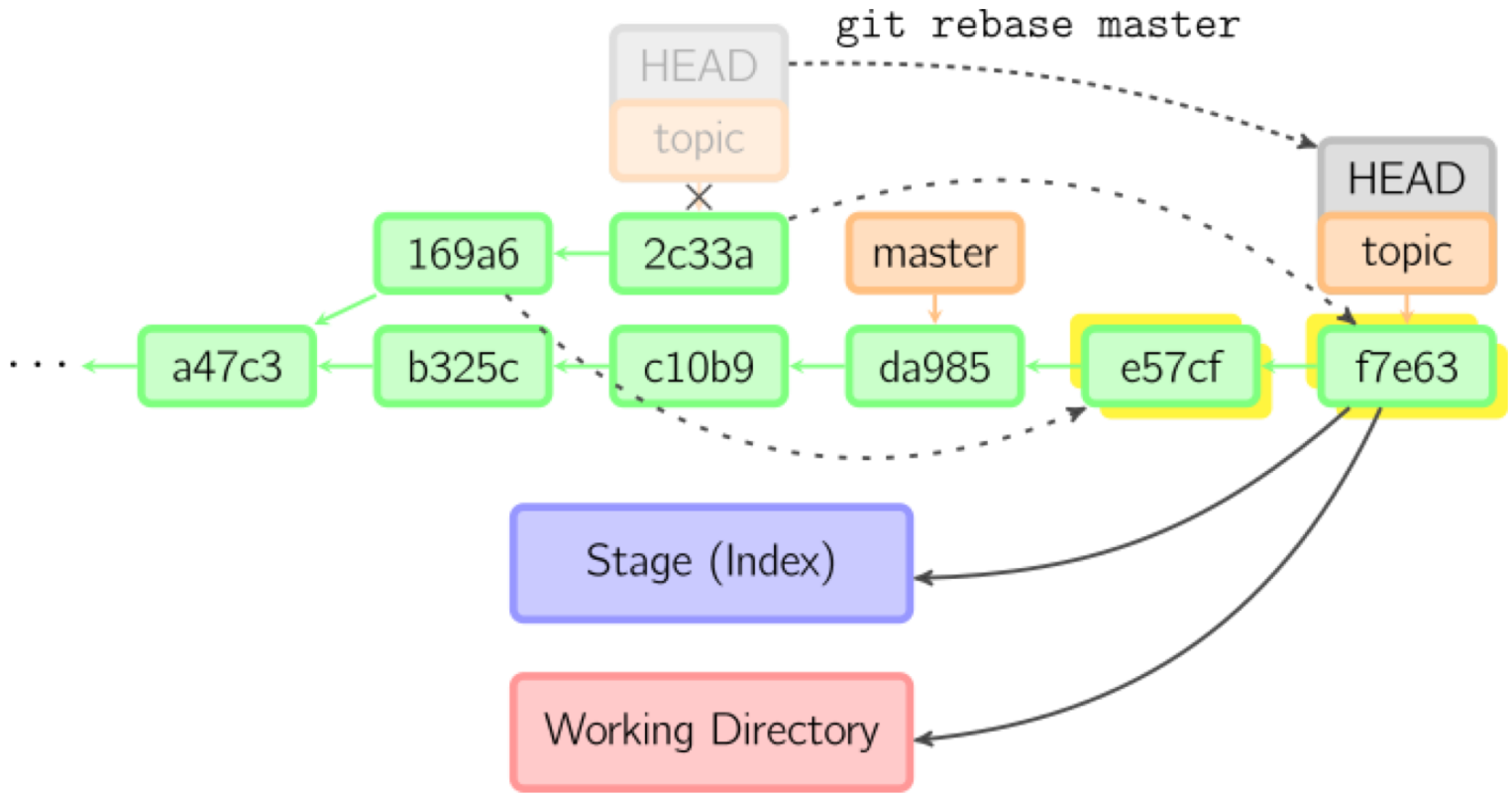


Combining Commands



- Add + Commit:
  - git commit –a
  - equivalent to running git add on all filenames that existed in the latest commit, and then running git commit
- Reset + Checkout:
  - git checkout HEAD -- <file> …
  - copies *files* from the latest commit to both the stage and the working directory

Digging Deeper (Self-Exploration)



https://marklodato.github.io/visual-git-guide/index-en.html

GitHub Basics

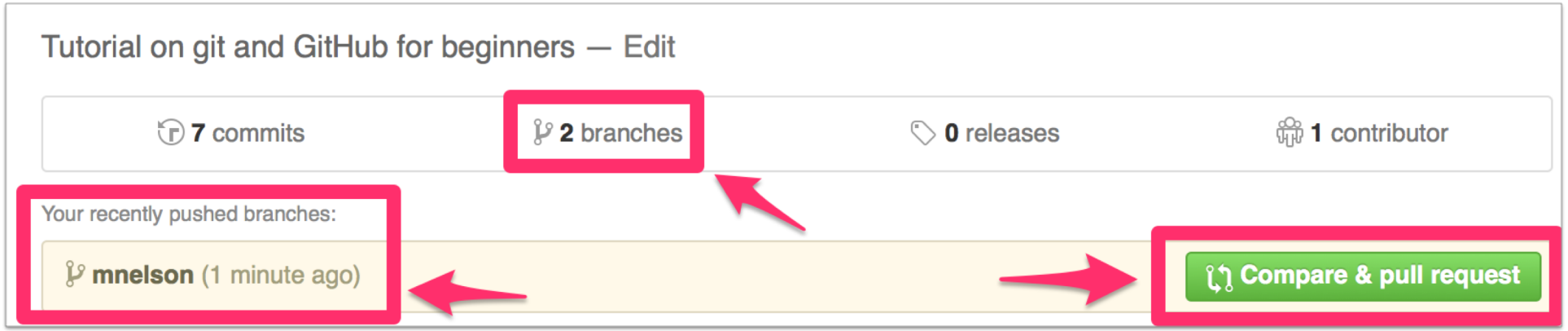
Material built from:
http://product.hubspot.com/blog/git-and-github-tutorial-for-beginners

GitHub

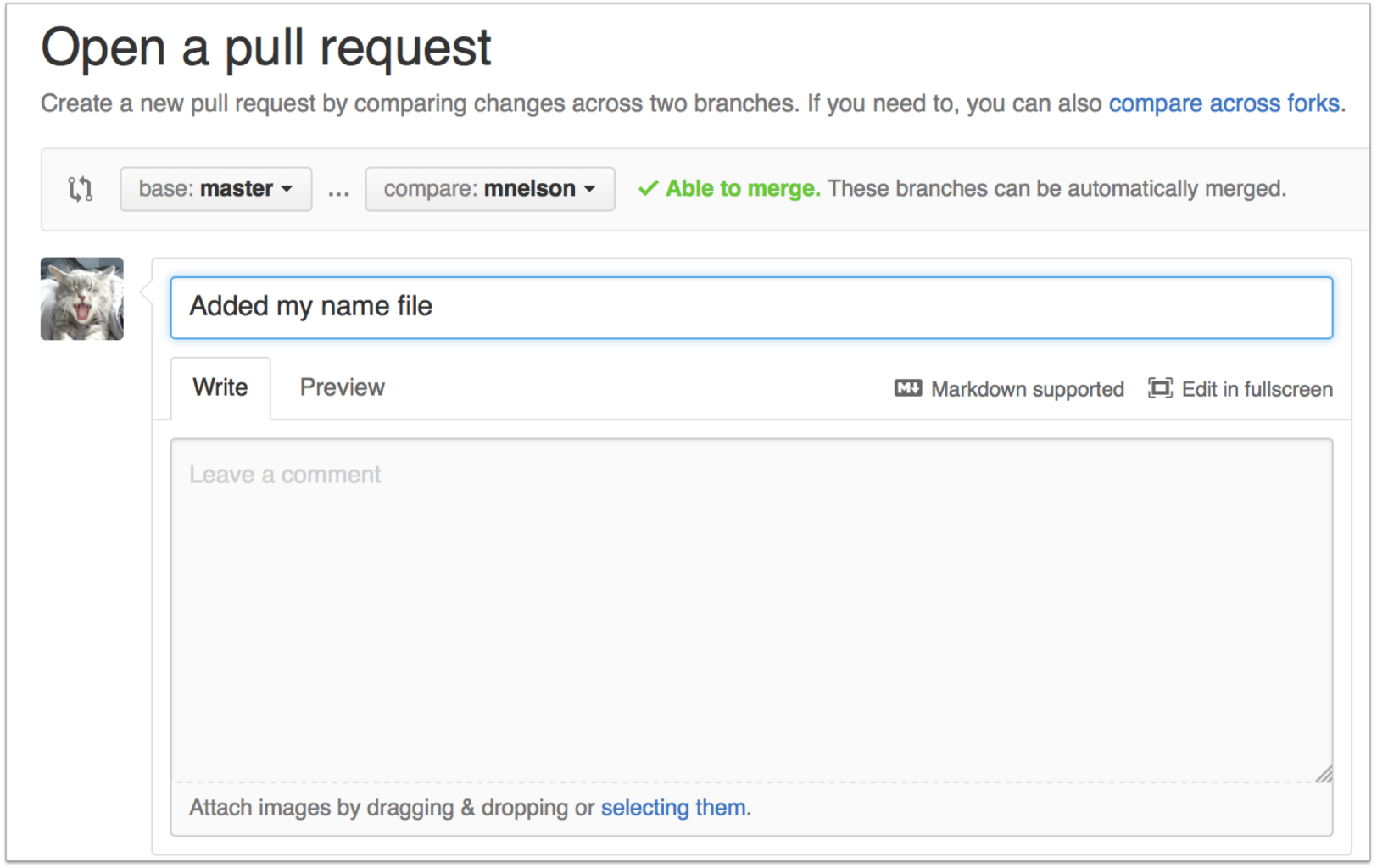
- Cloud hosted git repositories
  - Free for open-source projects & education
  - Paid for private repositories
- Can view commits, pull requests (PRs), and issues through your browser
  - PRs are formal requests for merges of commits
  - Issues are bug reports and GitHub provides discussion threads and notifications

PRs on GitHub

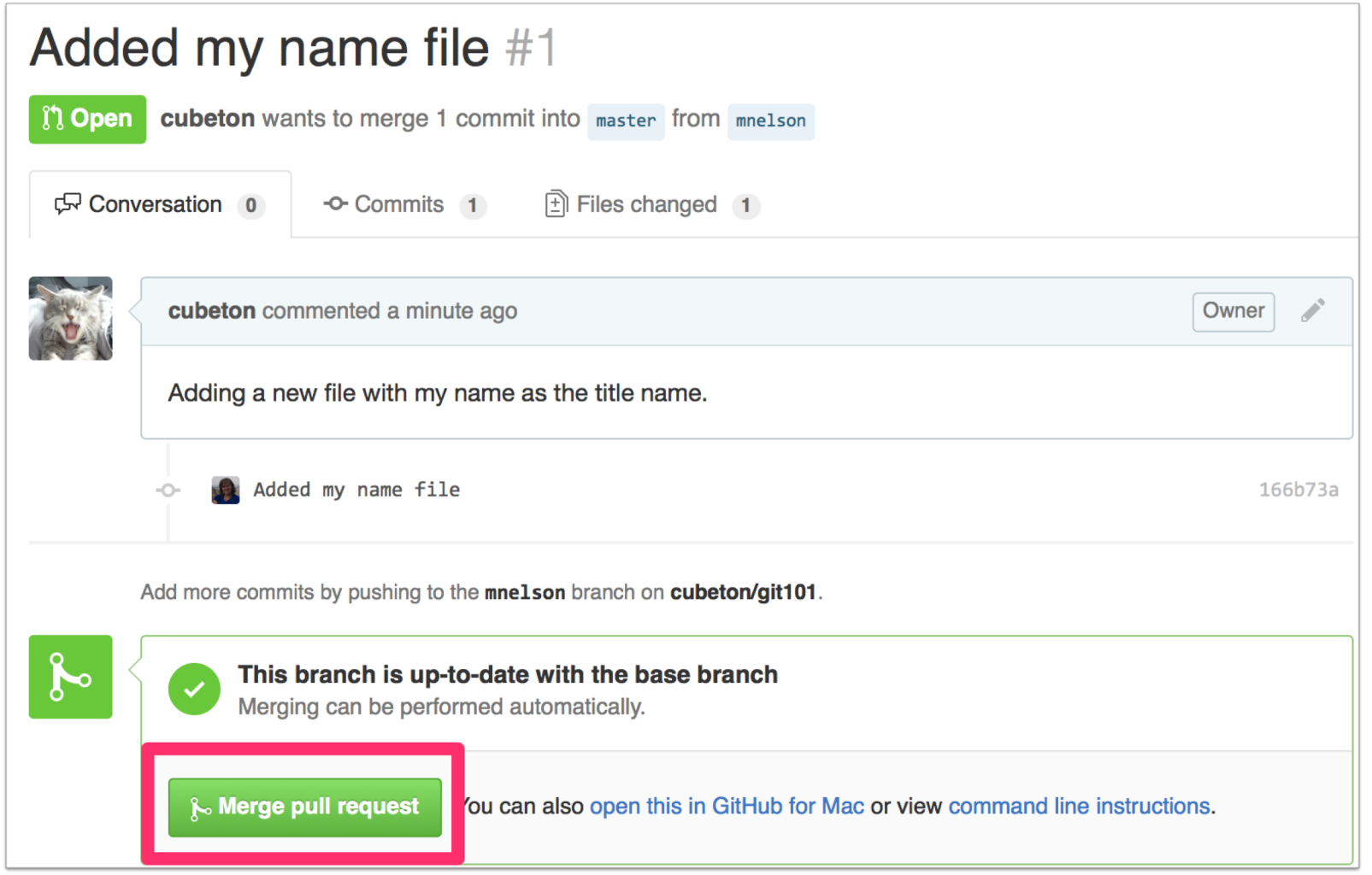
- Push your code
- Visit the repo on GitHub
- Click “compare & pull request”



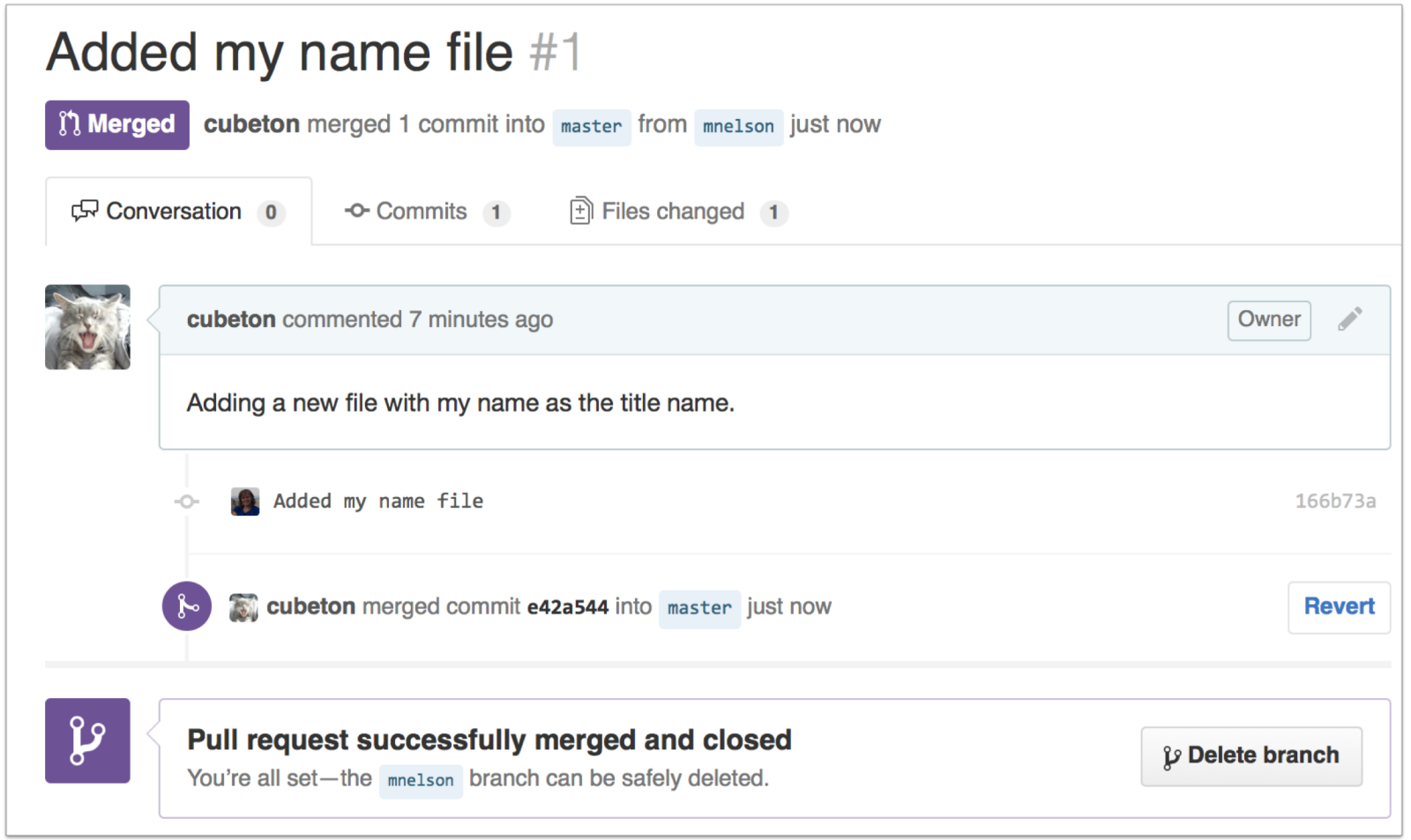
PRs on GitHub



PRs on GitHub



PRs on GitHub



Consolidate your expertise

Assignment 2

- Consolidate expertise in Python and Jupyter Notebook
- Learn to strategize on problems' solutions
  - Write a simple solution first
  - Use git commit to save significant steps
  - Use git branch for new solution efforts

Assignment 2

Problem 1:

- Write a function that finds the sum of the multiples of p and q below n
  - A problem may have multiple solutions, some more effective than others

Problem 2:

- Manipulate a list of names
  - Use the *csv module* to import a list of names
  - Score names in a list based on name “worth” and alphabetical order

Assignment 2

Problem 3:

- Find the smallest TPH number bigger than *n*
  - Triangular (T), Pentagonal (P), and Hexagonal (H) numbers are generated by given formulae
  - Write the code to find the next triangle number that is also pentagonal and hexagonal

a high-level introduction

NSF Funding Areas in HPC

Traditionally concentrated on enabling petascale capability
- Blue Waters – 13.3 petaflops, 2012
- Stampede – 9.6 petaflops, 2013
- Comet – ~2.0 petaflops, 2014

Has funded research into building clouds and computer science
- CloudLab
- Chameleon

Now funding clouds to do research
- Bridges (Hybrid system)
- Jetstream

From slides by Jeremy Fischer
jetstream-cloud.org/files/Jetstream-SIAM-CSE-Presentation.pdf

What is Jetstream and why does it exist?

- NSF's first production cloud facility
- Part of the NSF eXtreme Digital (XD) program
- Provides on-demand *interactive* computing and analysis
- Enables *configurable* environments and architectures
- User-friendly, widely accessible cloud environment
- User-selectable library of preconfigured virtual machines

From slides by Jeremy Fischer
jetstream-cloud.org/files/Jetstream-SIAM-CSE-Presentation.pdf

Cloud vs HPC

Adapting to a different environment:

- No reservations, no queueing
- More interactive use and less/no batch queuing
- What? No parallel filesystem?!?
- Being your own admin – hey, we have root!
- You really can have almost any (linux) software you want**
- Constantly getting new features (https://www.openstack.org/software/project-navigator/)

** Here there be dragons…

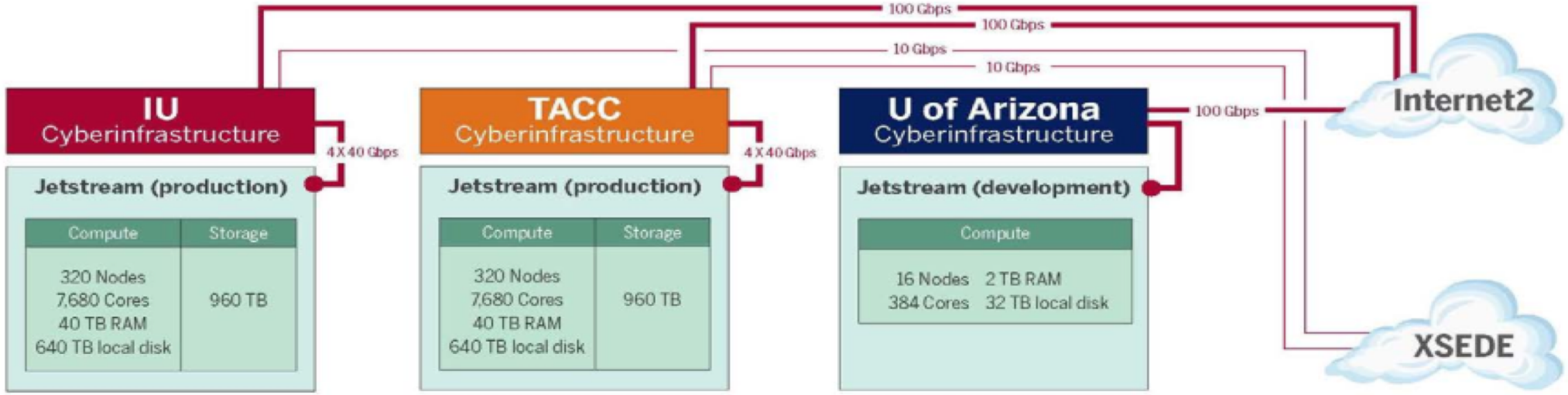
From slides by Jeremy Fischer
jetstream-cloud.org/files/Jetstream-SIAM-CSE-Presentation.pdf

Who uses Jetstream?

- The researcher needing a handful of cores (1 to 44/vCPU)
- Software creators and researchers needing to create their own customized virtual machines and workflows
- Science gateway creators using Jetstream as either the frontend or processor for scientific jobs
- STEM Educators teaching on a variety of subjects

From slides by Jeremy Fischer
jetstream-cloud.org/files/Jetstream-SIAM-CSE-Presentation.pdf

Jetstream System Overview



From slides by Jeremy Fischer
jetstream-cloud.org/files/Jetstream-SIAM-CSE-Presentation.pdf

Hardware and Instance "Flavors"

VM Host Configuration

- Dual Intel E-2680v3 "Haswell"
- 24 physical cores/node @ 2.5 GHz (Hyperthreading on)
- 128 GB RAM
- Dual 1 TB local disks
- 10GB dual uplink NIC
- Running KVM Hypervisor

Flavor	vCPUs	RAM	Storage	Per Node
m.tiny	1	2	8	46
m.small	2	4	20	23
m.medium	6	16	60	7
m.large	10	30	120	4
m.xlarge	24	60	240	2
m.xxlarge	44	120	480	1

- Short-term storage comes as part of launched instance
- Long-term storage is XSEDE-allocated
- Implemented on backend as OpenStack Volumes
- Each user gets 10 volumes up to 500GB total storage
- Piloting object storage as well after recent update

From slides by Jeremy Fischer
jetstream-cloud.org/files/Jetstream-SIAM-CSE-Presentation.pdf

Requesting access to Jetstream

- You can request startup allocations anytime. (Startups are simple!)
- You can request allocations for educational use anytime.

From slides by Jeremy Fischer
jetstream-cloud.org/files/Jetstream-SIAM-CSE-Presentation.pdf

- We'll send instructions for making an XCEDE account the week before we start using Jetstream in class.

- For now, back to Assignment 02!