Version control for researchers with Git and GitHub

Karin Knudson

karin.knudson@tufts.edu

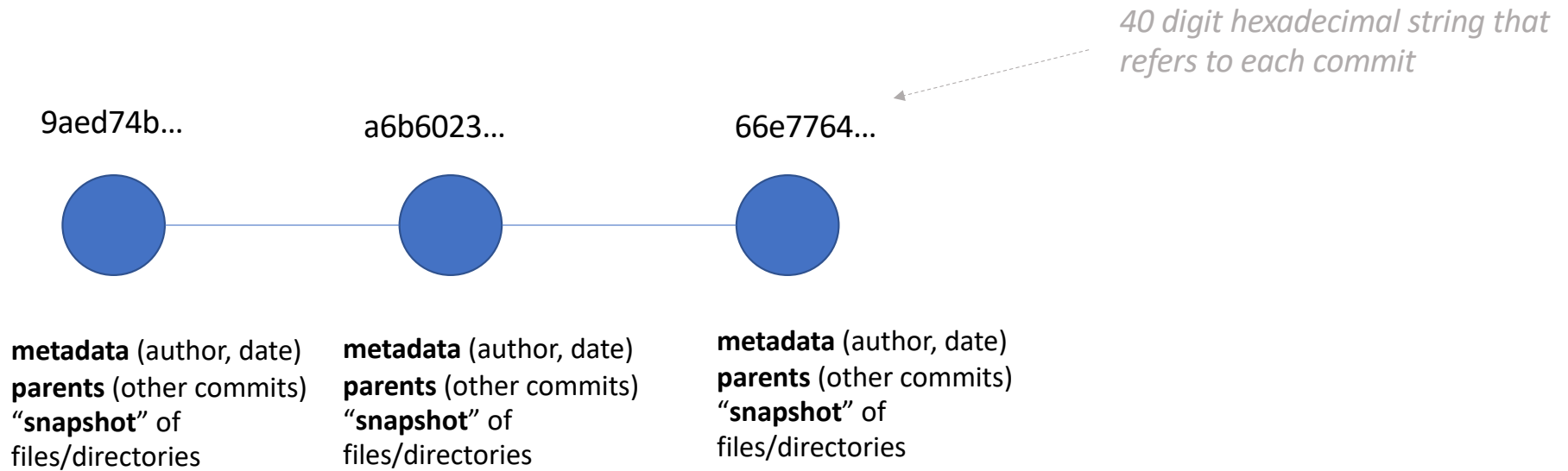Course materials: https://karink520.github.io/git-and-github-intro/

Why use Git and GitHub?

◆ **Sleep better** at night with version control and remote repositories

◆ **Collaborate** smoothly with teammates

◆ Promote and maintain **quality code**

◆ **Increase** the **impact** of your research

◆ Develop your **career**

◆ **Contribute** in the open source community
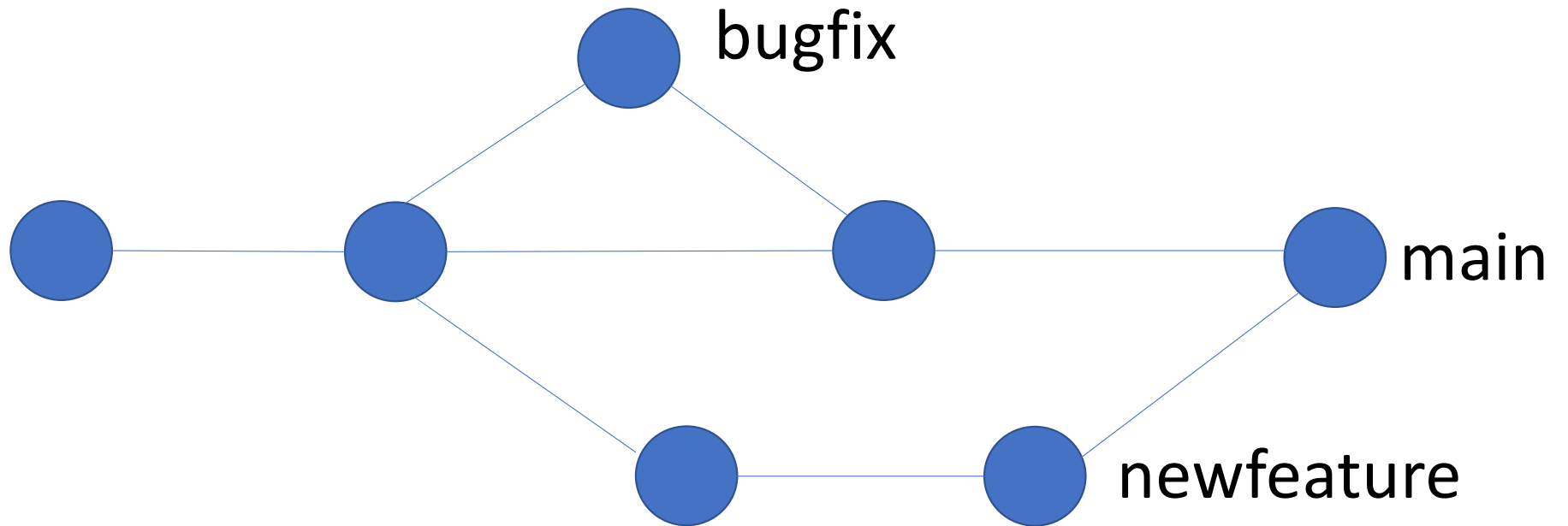
After this workshop you should be able to…

- Explain how version control, Git, and GitHub can help you
- Create a new project that is tracked with Git, or add Git to an existing project
- Use a simple workflow with Git and GitHub for small or individual projects
- Use a more complex workflow with branches and pull requests
- Contribute to some else's open source project on GitHub
- Know where to go to learn more

A few key concepts

commits

9aed74b...

a6b6023...

66e7764...

40 digit hexadecimal string that refers to each commit

metadata (author, date)
parents (other commits)
"**snapshot**" of
files/directories

metadata (author, date)
parents (other commits)
"**snapshot**" of
files/directories

metadata (author, date)
parents (other commits)
"**snapshot**" of
files/directories

branching

bugfix

main

newfeature

Staging area

We may not want to commit *all* of files or changes. We will ***add*** files and changes to a ***staging area*** before we commit them.

Git vs. GitHub

- Git = software for version control

Will learn to use basic Git commands: `init, remote, fetch, merge, status, add, commit, merge, push, fetch, checkout`

- GitHub = a repository hosting service with a graphical interface and additional tools for collaboration, etc.

Will learn to put repositories on GitHub, collaborate on others' GitHub repositories, use GitHub pull requests.

Other options: e.g. BitBucket

Set up a new repository with Git and GitHub

- Initialize with `git init` (double check that this worked with `ls —a` to see the new directory called `.git` that this command created)

- Add and commit any files you want as starting points:

    `git add <filename>`
    `git commit —m "initial commit"`

- Connect your repository to a remote GitHub repository with GitHub's interface and `git remote add`

- Copy the content you created to your remote repository (hosted on GitHub) with `git push`
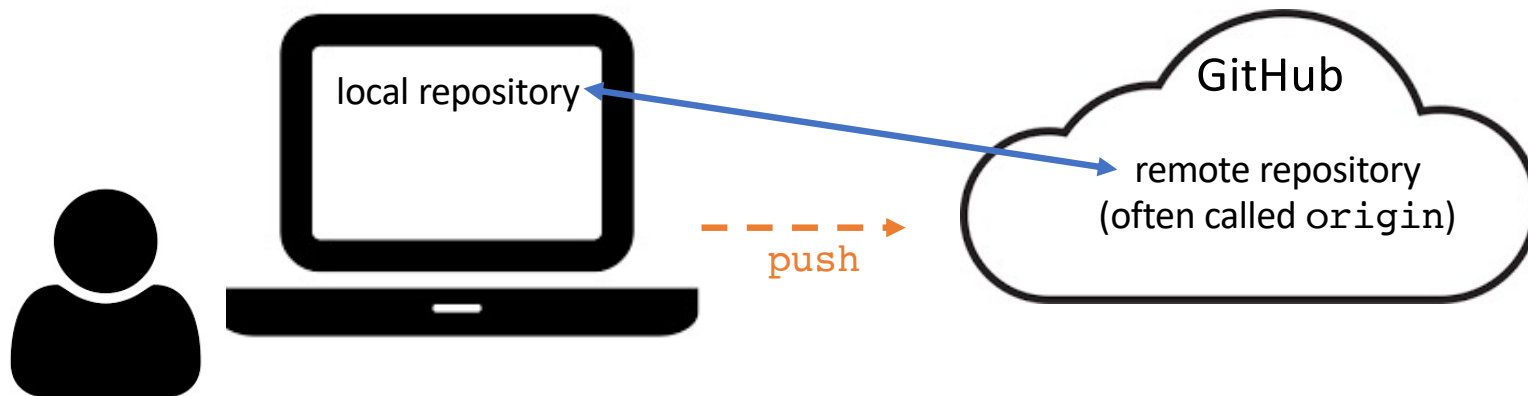
Follow the **numbered steps** in parts I and II

git + command + flags/arguments
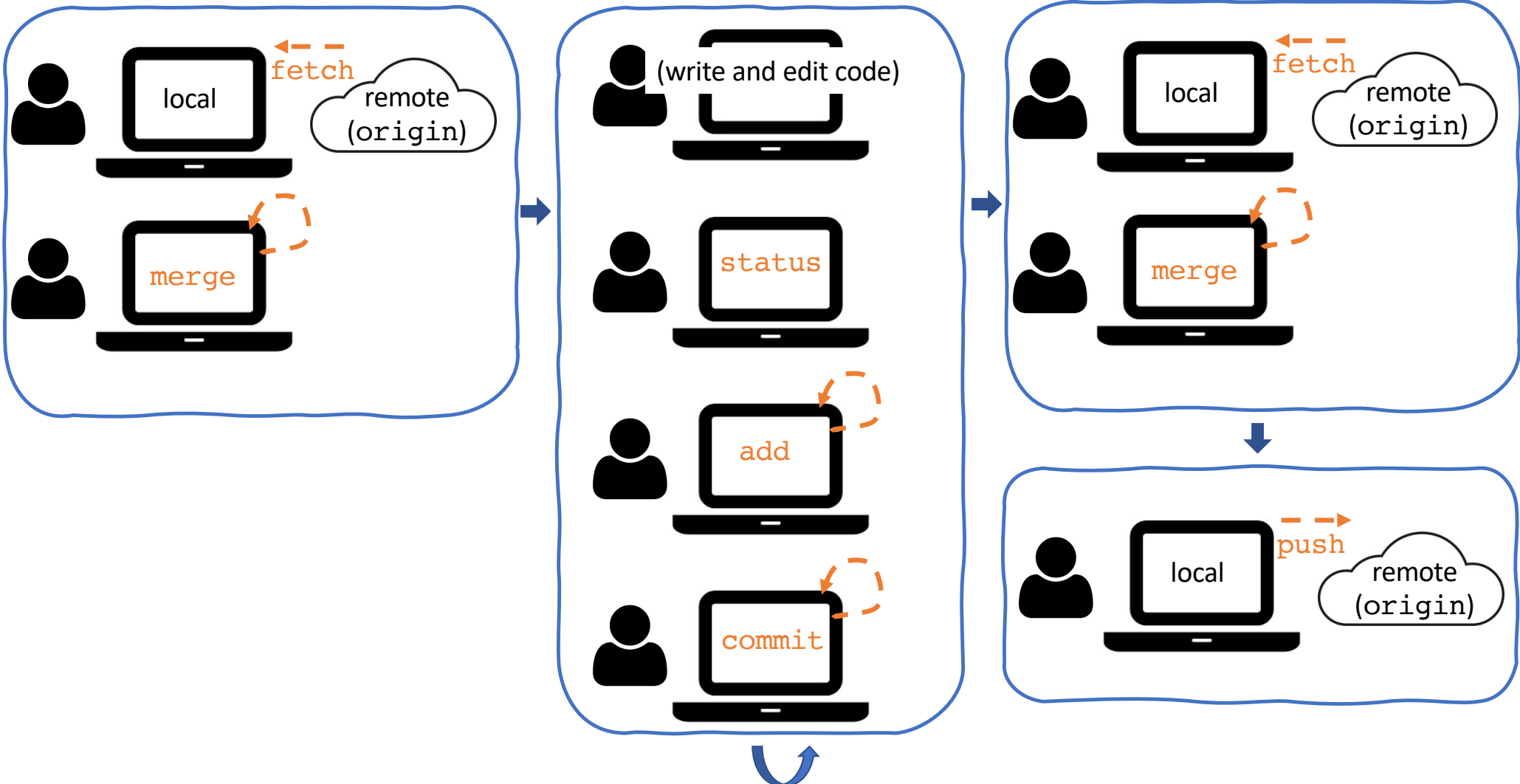
```
git fetch origin
git merge origin main
git add hello.py
git commit —m "my first commit"
git push -u origin main
```

Follow the instructions in part III to practice the workflow now

https://xkcd.com/1296/

Merge conflicts

- If you and a collaborator are simultaneously changing different parts of the code and merging, no problem!

- If you change a line of code, and in the meantime some one has made a different change to **the same line** and pushed those changes, you can have a merge conflict.

Checking out older commits

- Make sure you have committed all your changes, and then type something like

    `git checkout 66e77`

(numbers/digits refer to a commit – you can see the hashes for each commit by typing `git log` (or on GitHub).
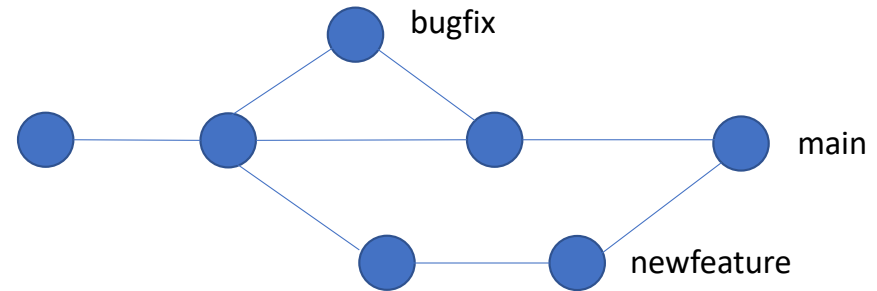
- Your files will change to the state they were in for the commit you just checked out. You will then be working in a detached HEAD state, and can look around and explore.

- Move back to where you were working (e.g. the main branch) to continue developing and editing with:

    `git checkout main`

(CAUTION: `git checkout <filename>` is dangerous).

Use Git branches and GitHub pull requests

- Instead of making all changes to main branch, create different branches for different features.
- Make branches locally, and then create and connect them to corresponding remote branches.
- Once your feature-specific branch is where you want it to be, *then* merge the changes on this branch back into the main branch of the remote repository.
- Use GitHub's **pull requests** to get collaborator's consent and input before merging the code on feature branch into the main branch of the remote repository.
- Move between branches with the `checkout` command

  (caution: `git checkout <filename>` is dangerous).

Follow the instructions in part V to practice the workflow now
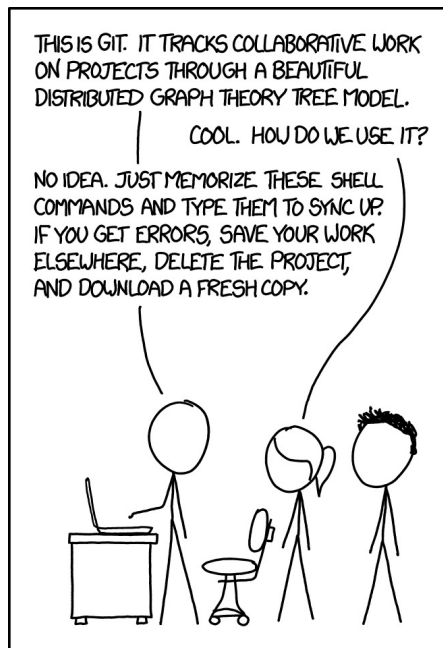
Collaborating on someone else's repository

Collaborating closely?

Someone can invite you as a collaborator to a repository on GitHub (Settings -> Manage Access -> Invite a collaborator), and then you can **clone** it (will get complete history).
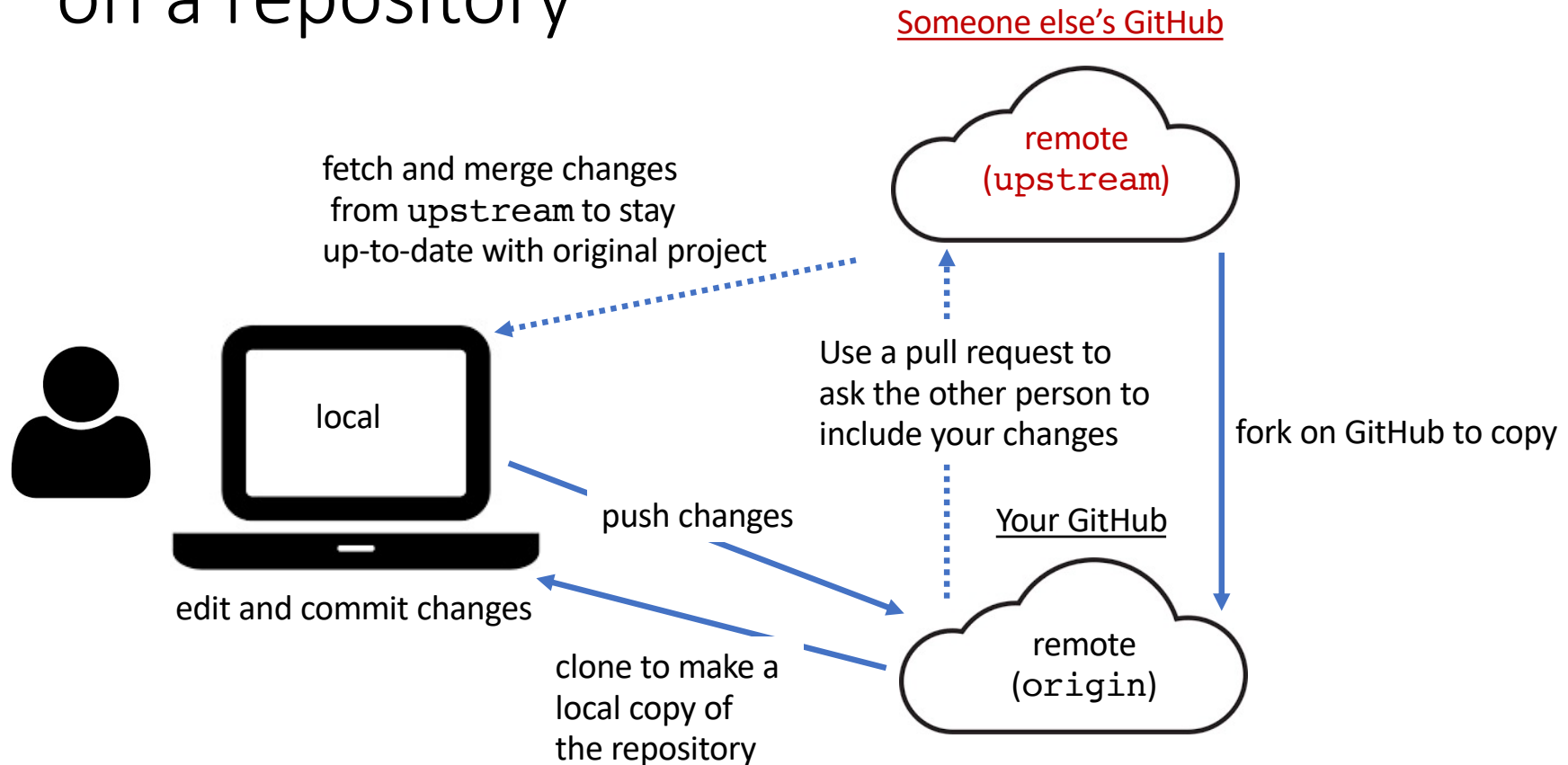
Just want someone's code (and no history)?

Download it

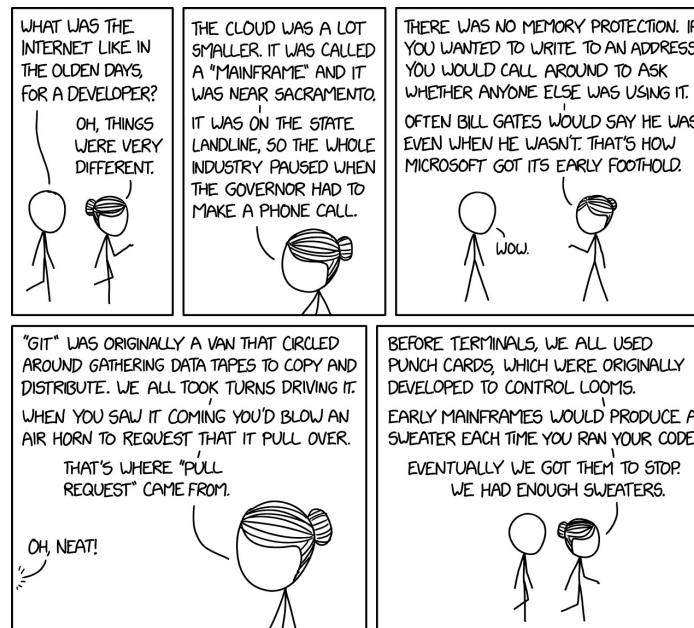*Contributing to a repository you don't have access to?*

Fork and **clone** (see next slide), and then use **pull requests**

Using GitHub forks and pull requests to build on a repository

Someone else's GitHub

remote
(upstream)

fetch and merge changes
from upstream to stay
up-to-date with original project

local

Use a pull request to
ask the other person to
include your changes

fork on GitHub to copy

push changes

Your GitHub

edit and commit changes

clone to make a
local copy of
the repository

remote
(origin)

"Homework"

- Contribute to
  https://github.com/karink520/TuftsGitHubSampleToUpdate using the
  process above as outlined in part VI.

https://xkcd.com/2324/

Some other important topics

- Stashing changes
- Undoing changes and reverting
- See what has changed with `git diff`
- Ignoring files you don't want to track
- Use ssh to connect to GitHub
- GitHub actions (e.g. to automatically run tests or other checks)

(see part VII)

Resources (part VIII)

- Searches and StackOverflow

- DangitGit!? https://dangitgit.com/

- GitHub Guides https://guides.github.com/introduction/git-handbook/

- Browser game for learning about Git branching https://learngitbranching.js.org/

- "A minimal tutorial": https://kbroman.org/github_tutorial

- Atlassian tutorials https://www.atlassian.com/git/tutorials

 and Git "cheat-sheet" https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet)

- MIT CSAIL's "Missing Semester" lesson on Git:

https://missing.csail.mit.edu/2020/version-control/

- Pro Git (book) https://git-scm.com/book/en/v2

- Renaming the default branch: https://dev.to/rhymu8354/git-renaming-the-master-branch-137b

Questions?