98-174 S17
Modern Version Control with Git

Ilan Biala (ibiala@andrew.cmu.edu)
Aaron Perley (aperley@andrew.cmu.edu)
https://www.andrew.cmu.edu/course/98-174/

Why should you take this course?

9. Know Git well.

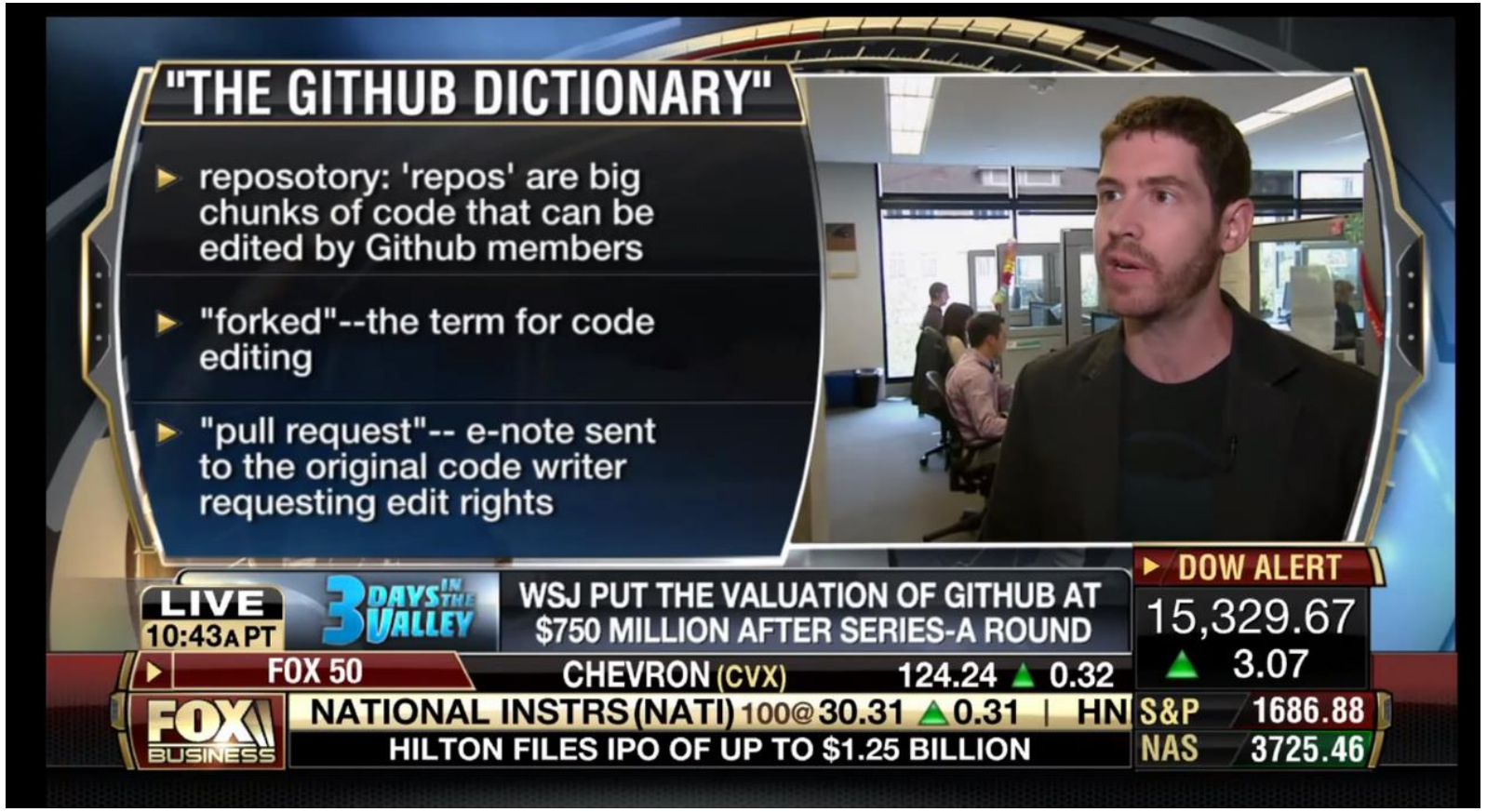
I realize that Git is more prevalent in some development communities over others, but Git is more than just a VCS (version control system). Because of its efficiencies in branching, it enables a very effective new flow that can be leveraged by both individuals and teams.

"Version control software is an essential part of the every-day of the modern software team's professional practices."

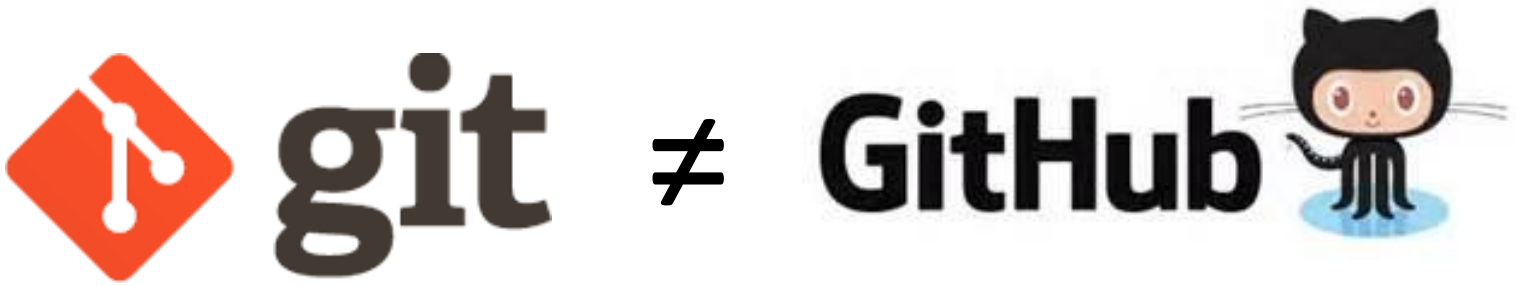
-- Atlassian Git Tutorial

Why should you take this course?

From a 2013 Fox News report:



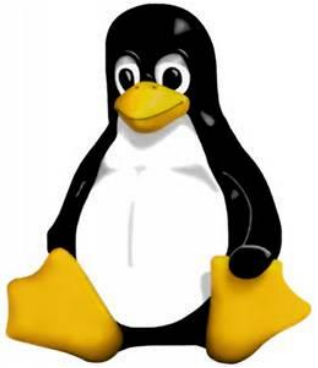
Git ≠ Github



What this course isn’t

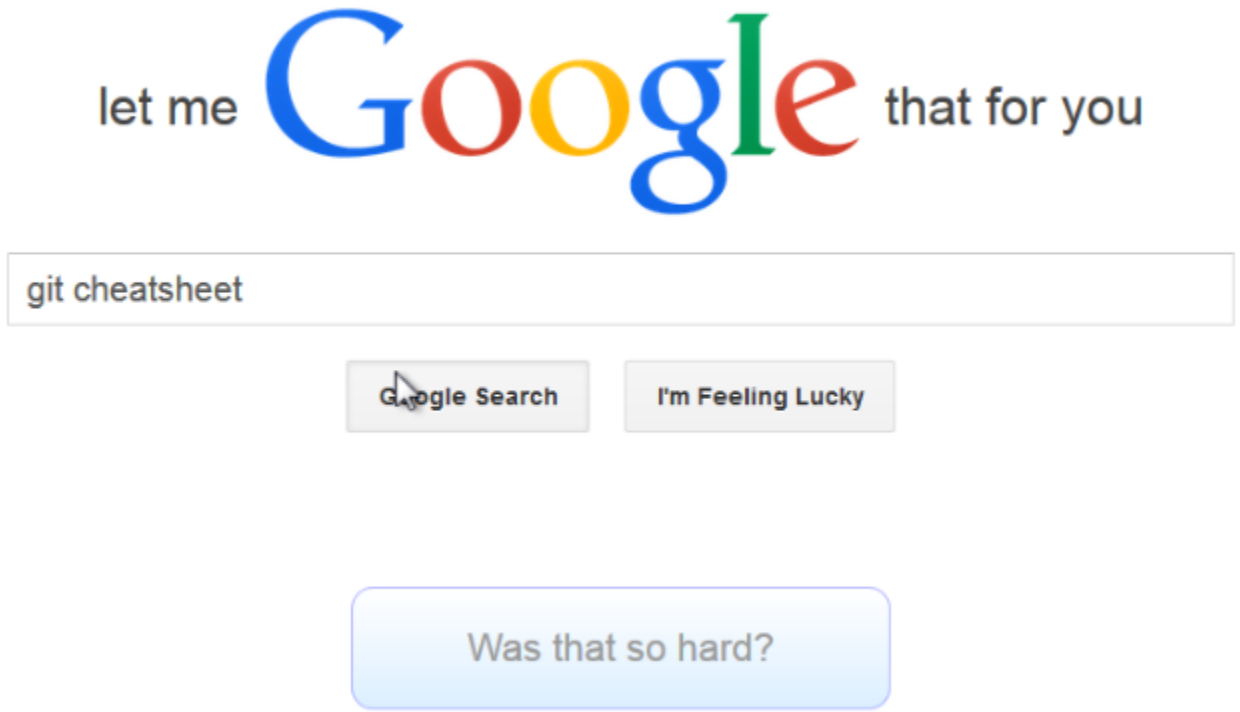
- For seasoned Linuxbeards





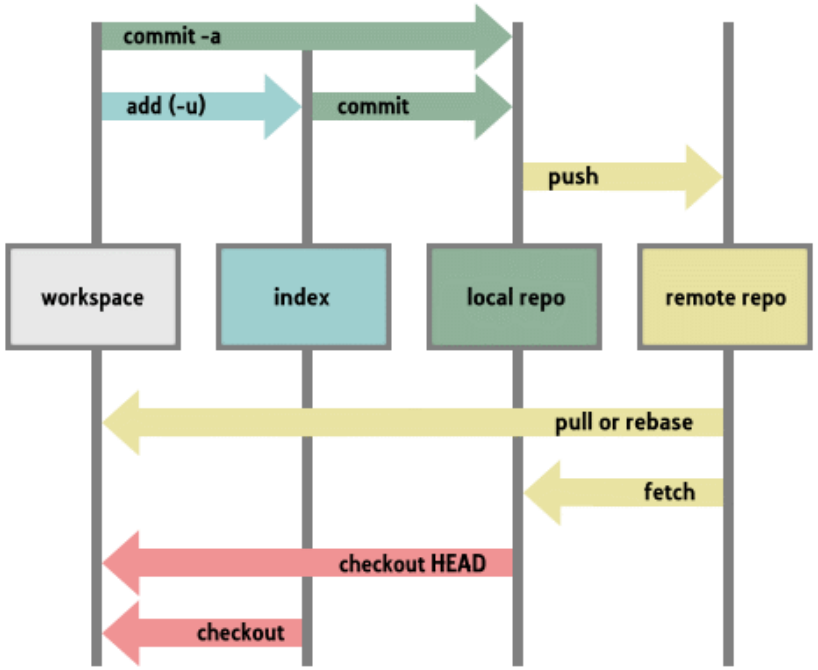

What this course isn’t

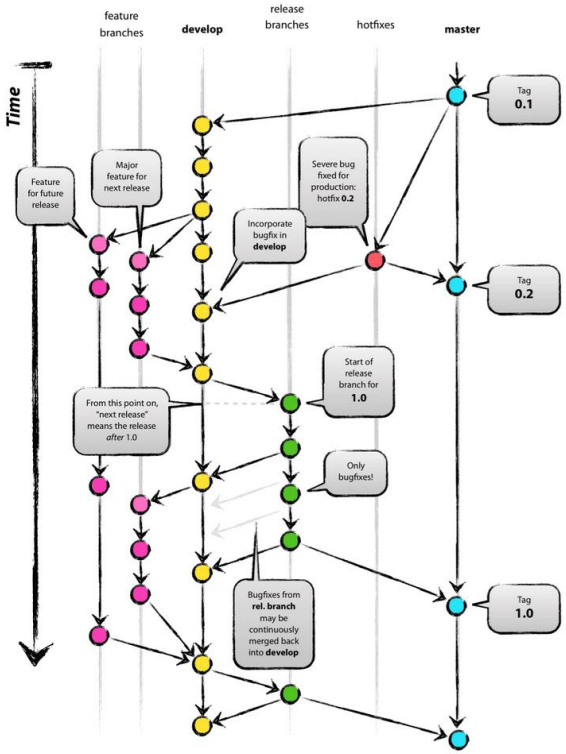
- A crashcourse on git commands



What this course **is** about

- Forming a **mental model** for how git interacts with versions of files
- Understanding how to use git (and a bit of Github) in a collaborative setting





Last Semester's Schedule

https://www.andrew.cmu.edu/course/98-174/f16/

Date	Lecture	Resources
8/29	Administrivia, Version Control Systems, and Git	Slides Installing Git Ry's Git Tutorial: Introduction git init git clone git log Homework 1
9/5	No Class (Labor Day)	
9/12	Simple Git Commits	Slides Git Book on the Staging Area More Reading on the Staging Area Online git Manpages Homework 2
9/19	Optional: Git Config and .gitignore	Git Book on git config Github Help on gitignore Sample .gitignore files from Github
9/26	More on Git Commits	Slides Git Book on Undoing Things (reset, checkout) Online git Manpages Worksheet / Homework 3
10/3	Branching and Merging	Slides Online git Manpages git dag alias Homework 4
10/10	Linus Torvalds on Git	Homework: Study for Midterm
10/17	Midterm	No Homework
10/24	No Class	No Homework, but review midterm solutions, see email
10/31	Remotes	Slides Online git Manpages Homework 5
11/7	No Class	Continue working Homework 5
11/14	Miscellaneous Git Commands	Slides Homework 6
11/21	Rebase	Slides No Homework - Enjoy Thanksgiving (Might post a review on rebase later this week)
11/28	Github	Slides In-class exercise copied from GPI Study for Final
12/5	Final	Good luck on your final exams!

Course Website

https://www.andrew.cmu.edu/course/98-174/

Grade Breakdown

Pass/No Credit, like every StuCo. To pass, get 70% out of:

- 20% Weekly Lecture Attendance
  (Tuesdays 6:30PM – 7:20PM, Baker Hall 140F)
- 30% Submitted work (often in-class)
- 20% Midterm (Date TBA)
- 30% Final (Date TBA)

More Course Details

- Prerequisite: Basic Unix Survival
- 3 Free Elective credits
- No official textbook, but we recommend Pro Git by Scott Chacon (free, online)
- No office hours unless specifically requested
  - Email Ilan and Aaron if you have questions
- Slides and lecture notes posted online

Course Policy

- By StuCo Policy, students with more than 2 unexcused absences must be given a No Pass in the course. Thus, email us if you're going to miss class for a legitimate reason, and you might get an excused absence.
- More than 15 minutes late = unexcused absence. Don't be late.
- Academic integrity applies. Don't cheat.
- No late homework.

Waitlist

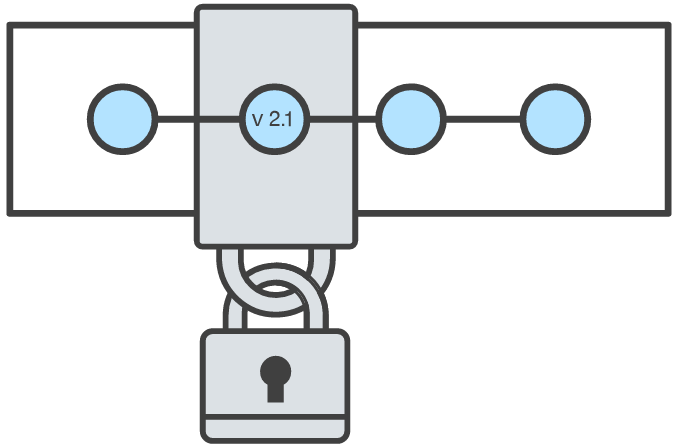
- If you are on the waitlist, please keep coming to class.

What is Version Control?

```
Aaron@HELIOS ~/112_term_project
$ ls
termproject_actually_final  termproject_v10  termproject_v3
termproject_final           termproject_v11  termproject_v4
termproject_handin          termproject_v12  termproject_v5
termproject_old_idea        termproject_v13  termproject_v6
termproject_superfrogger    termproject_v14  termproject_v7
termproject_temp            termproject_v15  termproject_v8
termproject_this_one_works  termproject_v16  termproject_v9
termproject_v1              termproject_v2
```

Goals of Version Control

- Be able to search through revision history and retrieve previous versions of any file in a project
- Be able to share changes with collaborators on a project
- Be able to confidently make large changes to existing files



https://www.atlassian.com/git/tutorials/what-is-version-control

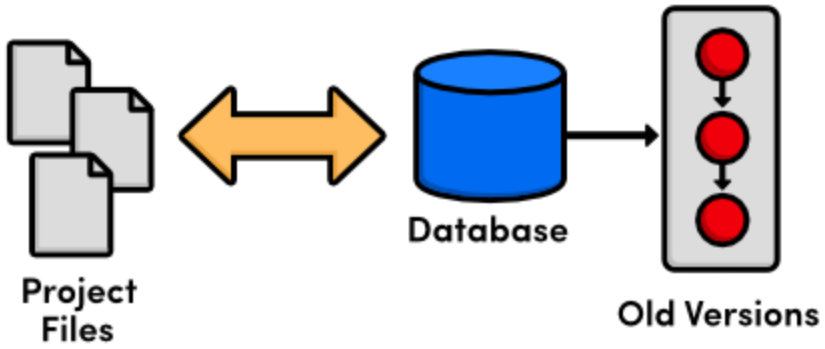
Named Folders Approach

```
Aaron@HELIOS ~/112_term_project
$ ls
termproject_actually_final  termproject_v10  termproject_v3
termproject_final           termproject_v11  termproject_v4
termproject_handin          termproject_v12  termproject_v5
termproject_old_idea        termproject_v13  termproject_v6
termproject_superfrogger    termproject_v14  termproject_v7
termproject_temp            termproject_v15  termproject_v8
termproject_this_one_works  termproject_v16  termproject_v9
termproject_v1              termproject_v2
```

- Easy
- Familiar
- …

- Can be hard to track
- Memory-intensive
- Can be slow
- Hard to share
- No record of authorship

Local Database of Versions Approach

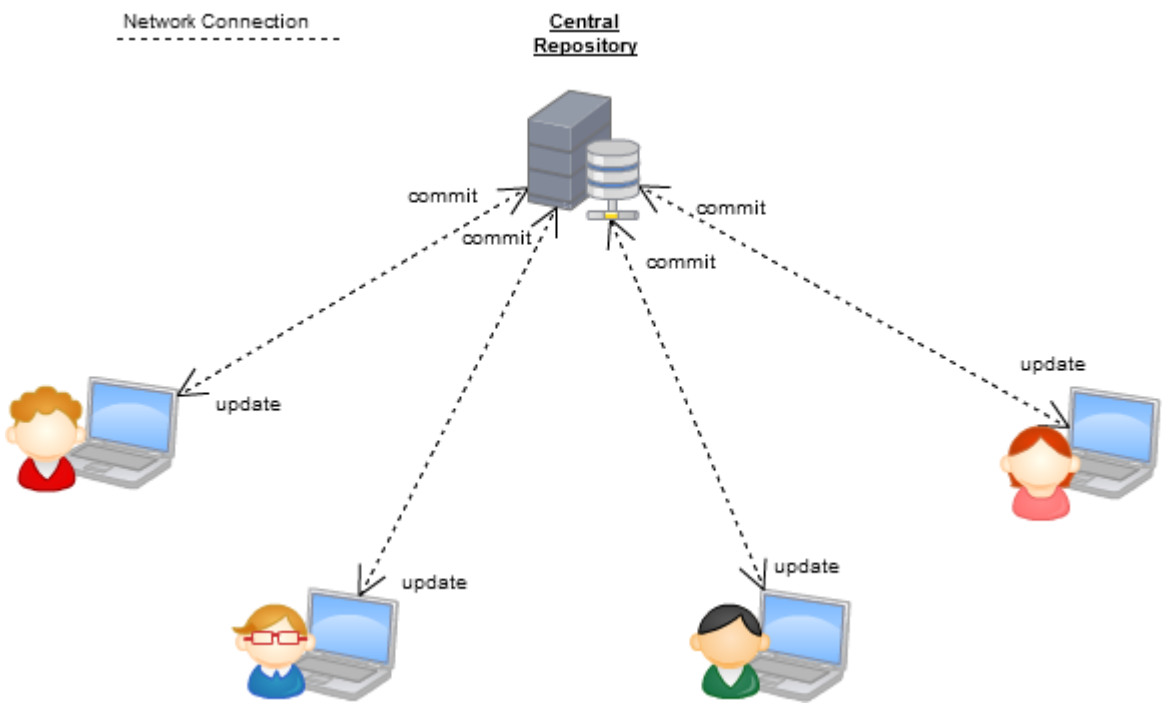


- Provides an abstraction over finding the right versions of files and replacing them in the project
- Can’t share with collaborators

Centralized Version Control Systems

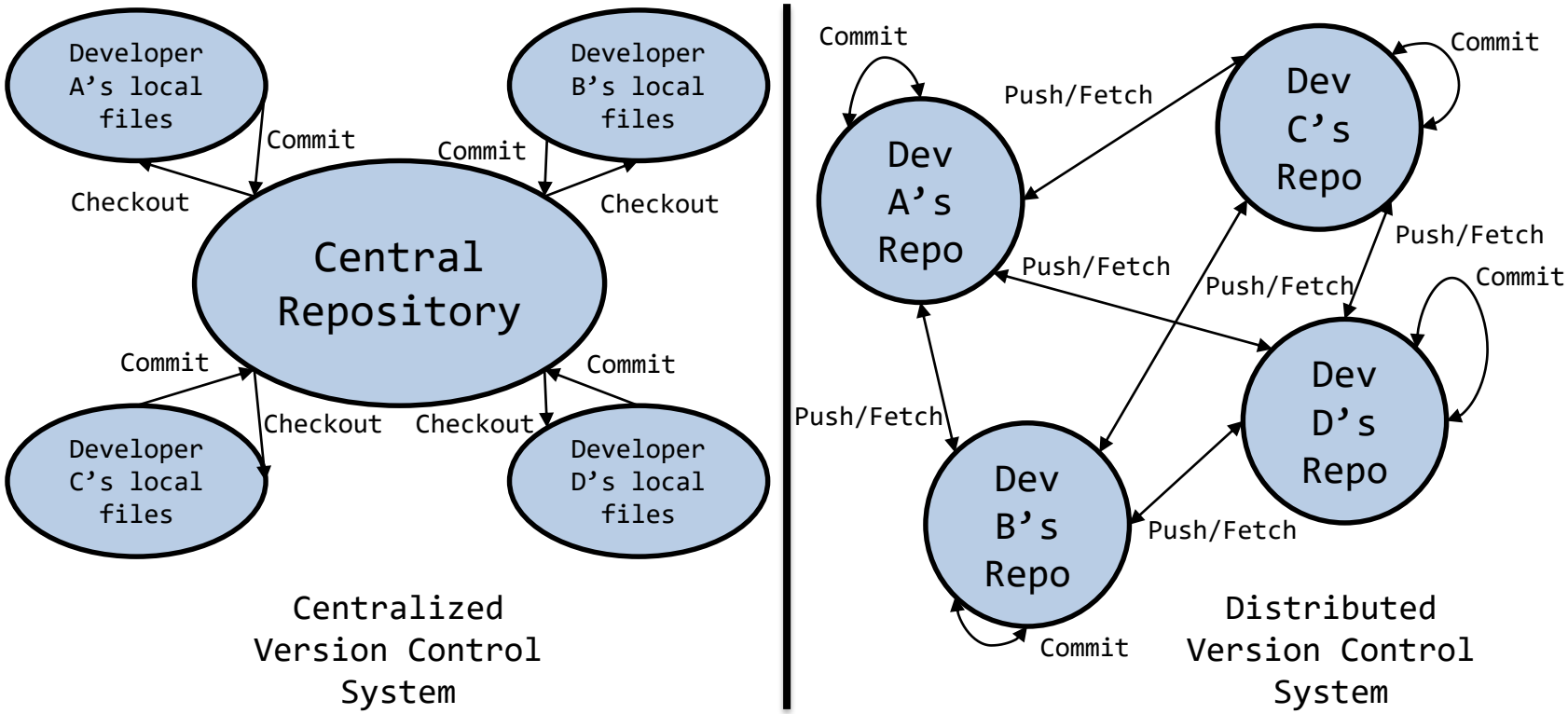
- A central, blessed repository determines the **order** of commits (“versions” of the project)
- Collaborators “push” changes (commits) to this repository.
- Any new commits must be compatible with the most recent commit. If it isn’t, somebody must “merge” it in.

- Examples: SVN, CVS, Perforce



Distributed Version Control Systems (DVCS)

- No central repository
- Every repository has every commit
- Examples: Git, Mercurial



Git

- Created in 2005 by Linus Torvalds to maintain the Linux kernel. Oh, and he created that too.
- Distributed VCS

https://www.git-scm.com/

Installing Git

https://www.andrew.cmu.edu/course/98-174/lecturenotes/installing_git.html