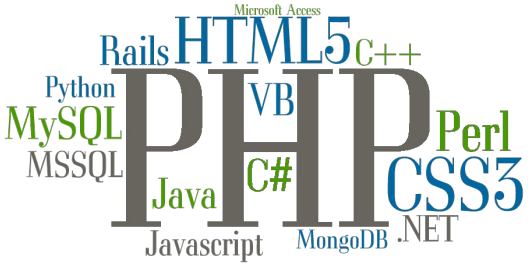
An overview of Version Control

Hossein A. Rahmani, Rasoul Mirzaei, Vahid Taghizad
Software Engineering, TAs

Fall 2018

python


Microsoft Access
Rails
HTML5
C++
Python
VB
MySQL
PHP
Perl
MSSQL
C#
CSS3
Java
.NET
Javascript
MongoDB

Team Working

Question

How do you share code?

Discussion

Working in a team, you need to share code – How?

- Email attachments?
- What if two people edit the same file at the same time?
- Google Docs or Dropbox?
  - No merging in Dropbox
  - Google Docs not meant for code
    - No syntax highlighting, etc.

Scenario #1

1. You change one part of a program (*it works*)
2. Your co-worker changes another part (*it works*)
3. You put them together (*it doesn't work!*)
4. Some change in one part must have broken something in the other part

What were all the changes?

Scenario #2

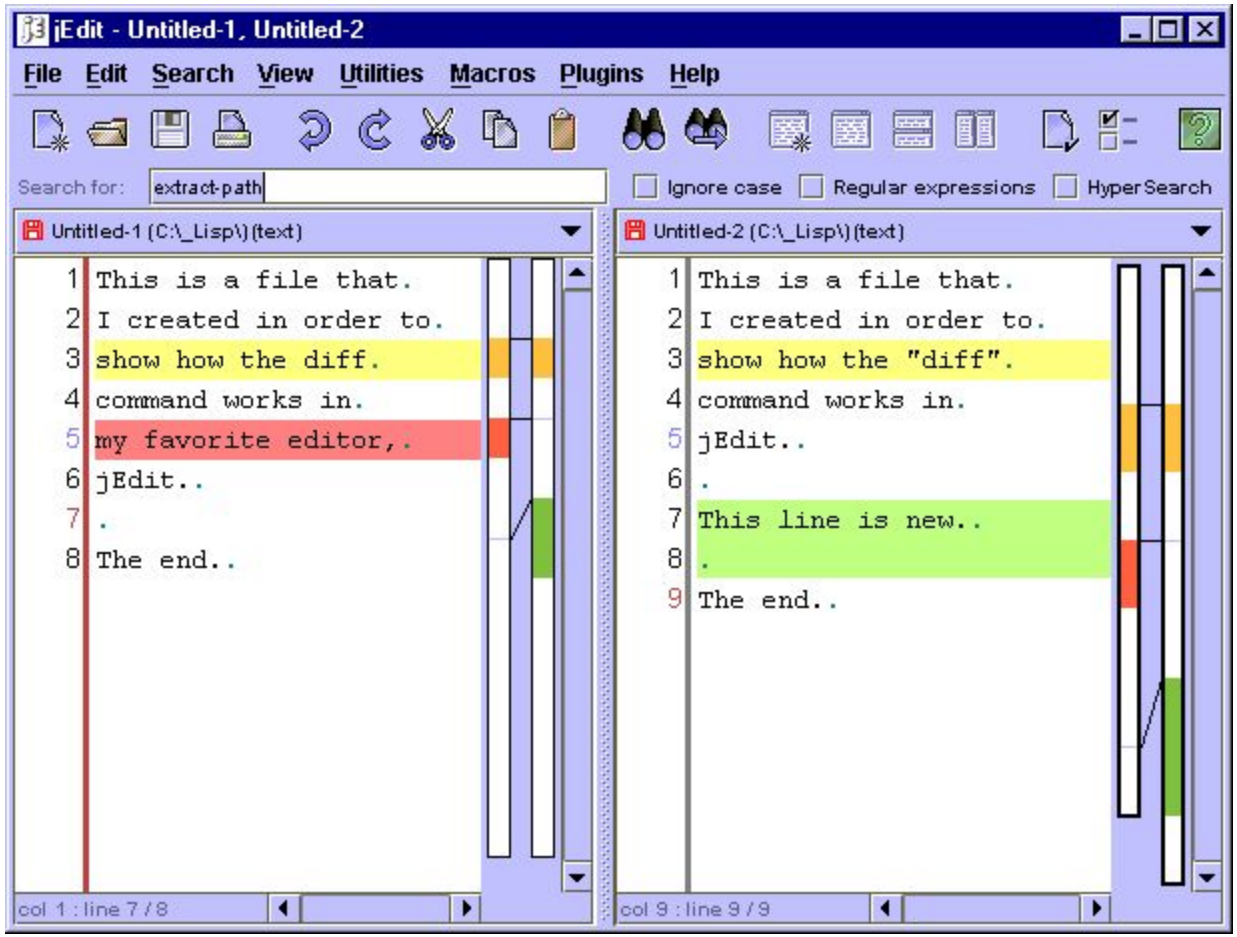
1. You make a number of improvements to a class
2. Your co-worker makes a number of ***different*** improvements to the ***same*** class

How can you merge these changes?

diff Tools

- There are a number of tools that help you spot changes (differences) between two files
- Tools include diff, rcsdiff, jDiff, etc.
- Of course, they won't help unless you kept a copy of the older version
- Differencing tools are useful for finding a *small* number of differences in a *few* files

diff Tools (Cont.)



Version Control System

A version control system (often called a source code control system) does these things:

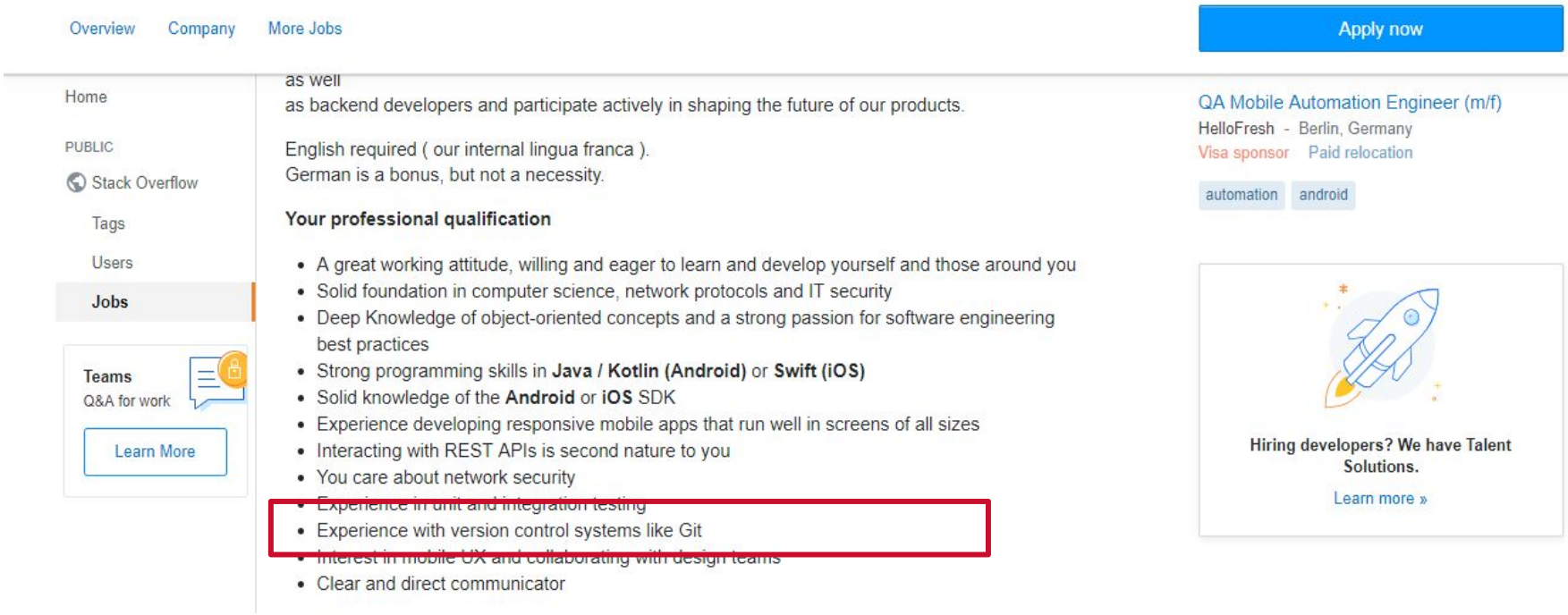
- Keeps multiple (older and newer) versions of everything (not just source code)
- Requests comments regarding every change
- Allows “check in” and “check out” of files so you know which files someone else is working on
- Displays differences between versions

Why Version Control?

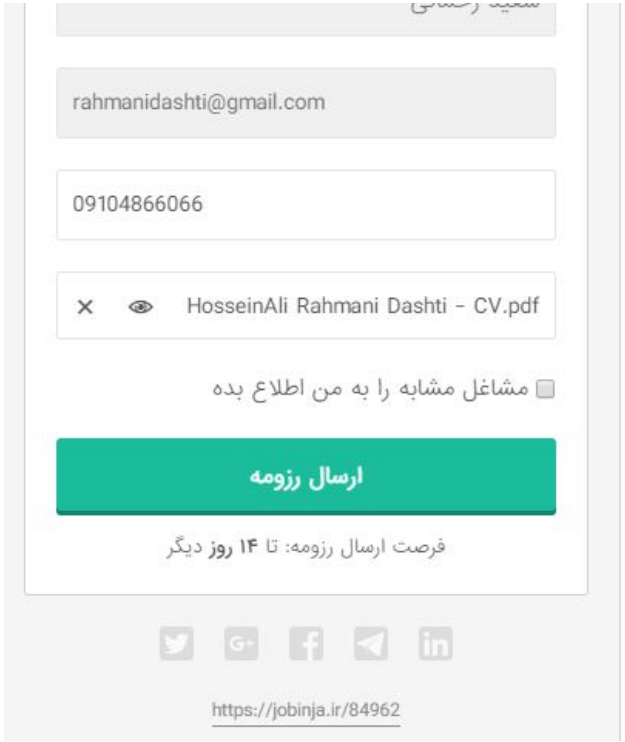
- It's a must in today's computing industry
- You can even version control documents
- Also called source control, code management, concurrent versioning, reversión control

- Advantages:
  - Track changes, keep logs of past development
  - Automatic backups
  - Allows teamwork

VCS in Job responsibilities (SO)



VCS in Job responsibilities (joninja)



شرح موقعیت شغلی

مهارت های عمومی

- روحیه کار تیمی
- توانایی درک مسئله و ارائه راه حل
- توانایی استفاده روان از متون انگلیسی تخصصی مرتبط
- علاقه به یادگیری و بروز بودن

مهارت های برنامه نویسی

- تسلط بر زبان برنامه نویسی PHP
- تسلط بر مفاهیم شی گرایی
- تسلط بر آخرین نسخه فریم‌ورک لاراول (5.7)
- تسلط بر مدیریت خطاها
- آشنایی با مفاهیم و ابزارهای توسعه وب (HTML، JavaScript، CSS، jQuery،...)
- آشنایی با PHP 7
- آشنایی با Git

Version Control System Types

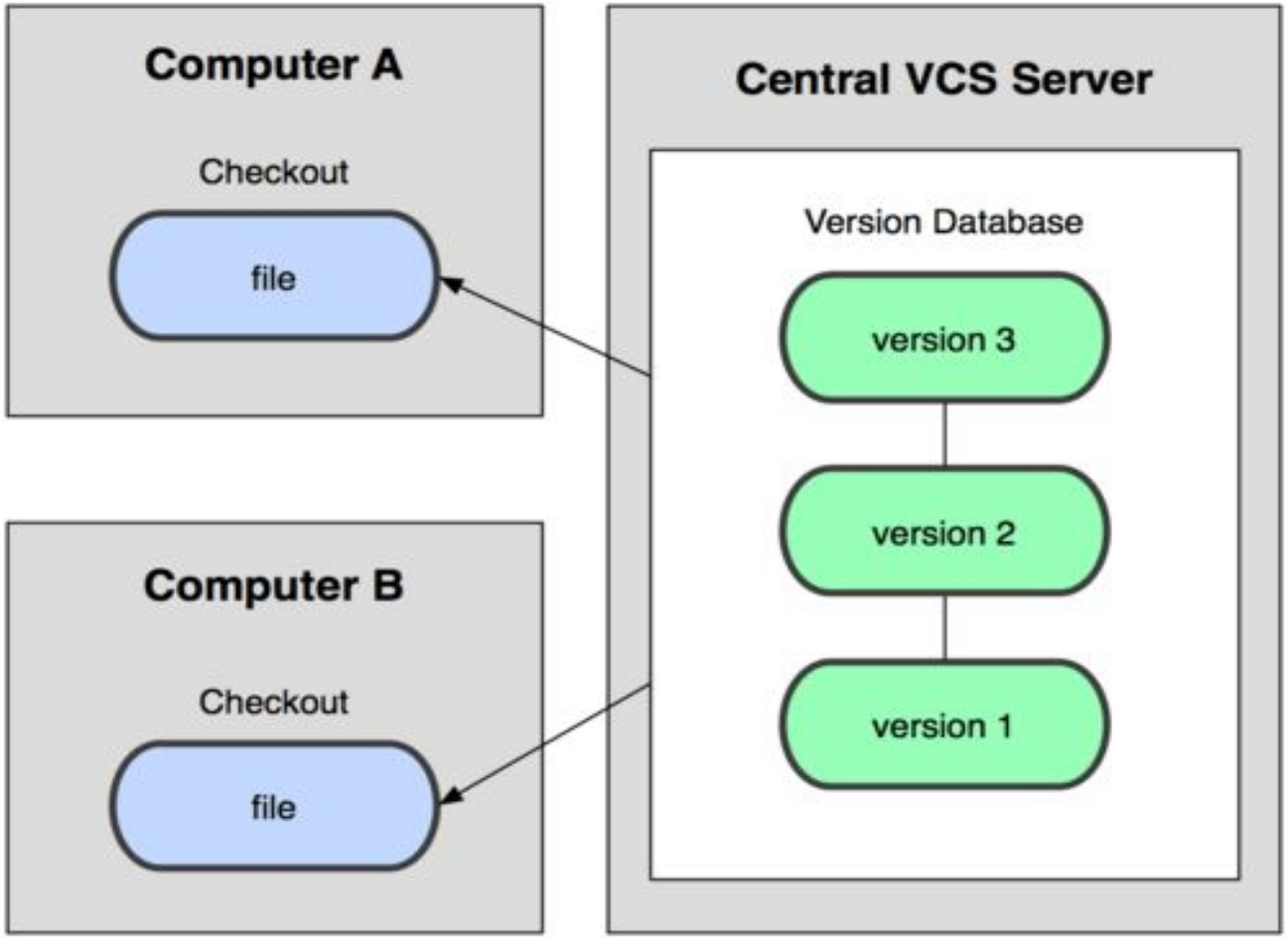
- Distributed VCS: **Git**, Mercurial
- Centralized VCS: CVS, Perforce, **SVN**

Version Control System: **Centralized**

The main difference between the two classes is that Centralized VCSs keep the history of changes on a central server from which everyone requests the latest version of the work and pushes the latest changes to. This means that everyone sharing the server also shares everyone's work. Sourceforge.net uses this type of versioning in their projects.

[https://www.oshyn.com/blogs/2012/june/version_control_systems__distributed_vs__centralized]

Centralized Version Control System



https://git-scm.com/book/en/v1/Getting-Started-About-Version-Control

Centralized Version Control System: **Subversion**

Subversion uses a centralized revision control model. **Ben Collins-Sussman**, one of the designers of Subversion, believes a centralised model would help prevent "insecure programmers" from hiding their work from other team members. Some users of version control systems see the centralised model as detrimental; famously, **Linus Torvalds** attacked Subversion model and its developers.

Limitation of Centralized Version Control

- Tell me! What do you think?

Limitation of Centralized Version Control

- Tell me! What do you think?

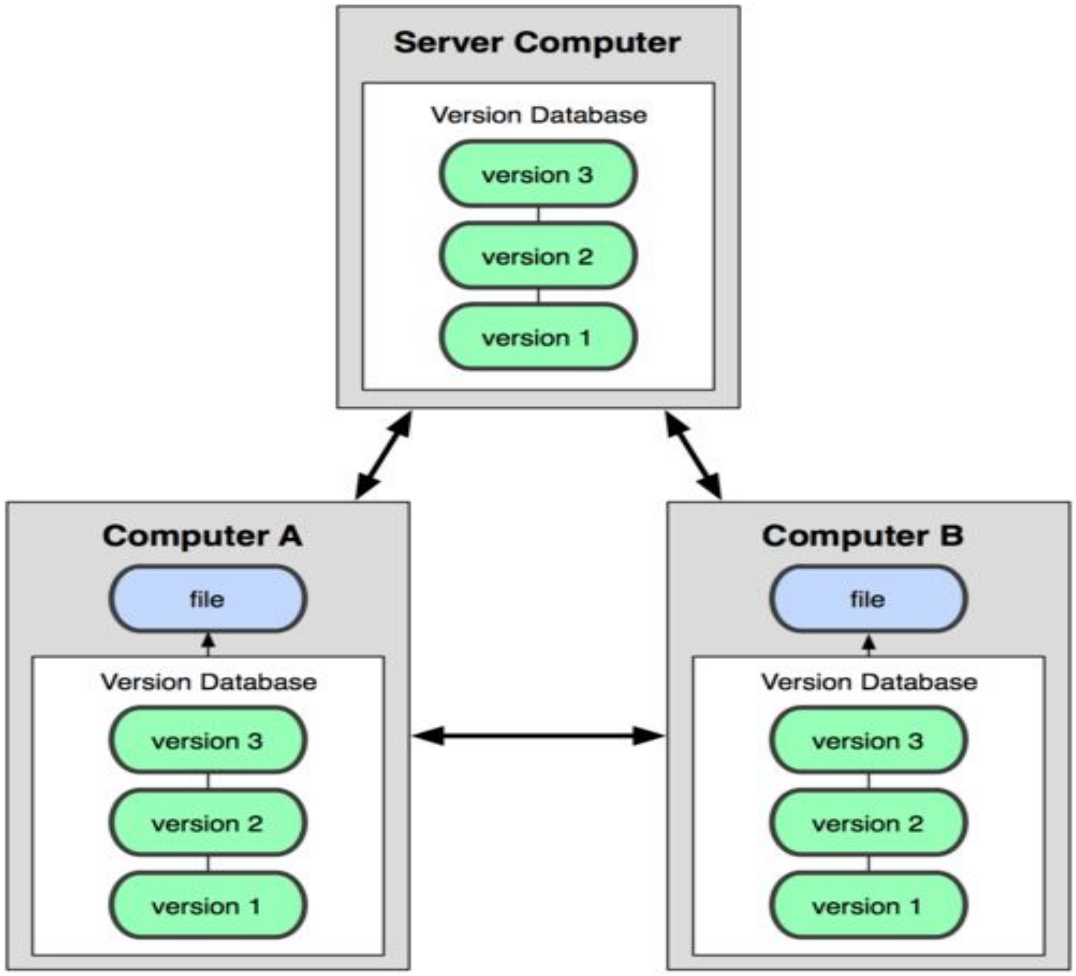
- If the main server goes down, developers can't save versioned changes
- Remote commits are slow
- Unsolicited changes might ruin development
- If the central database is corrupted, the entire history could be lost (security issues)

Version Control System: **Distributed**

On the other hand, on a Distributed VCS, everyone has a local copy of the entire work history. This means that it is not necessary to be online to change revisions or add changes to the work. “Distributed” comes from the fact that there isn’t a central entity in charge of the work’s history, so that anyone can sync with any other team member. This helps avoid failure due to a crash of the central versioning server. Open source projects, such as Mozilla Firefox, tend to use this type of versioning.

[https://www.oshyn.com/blogs/2012/june/version_control_systems__distributed_vs__centralized]

Distributed Version Control System



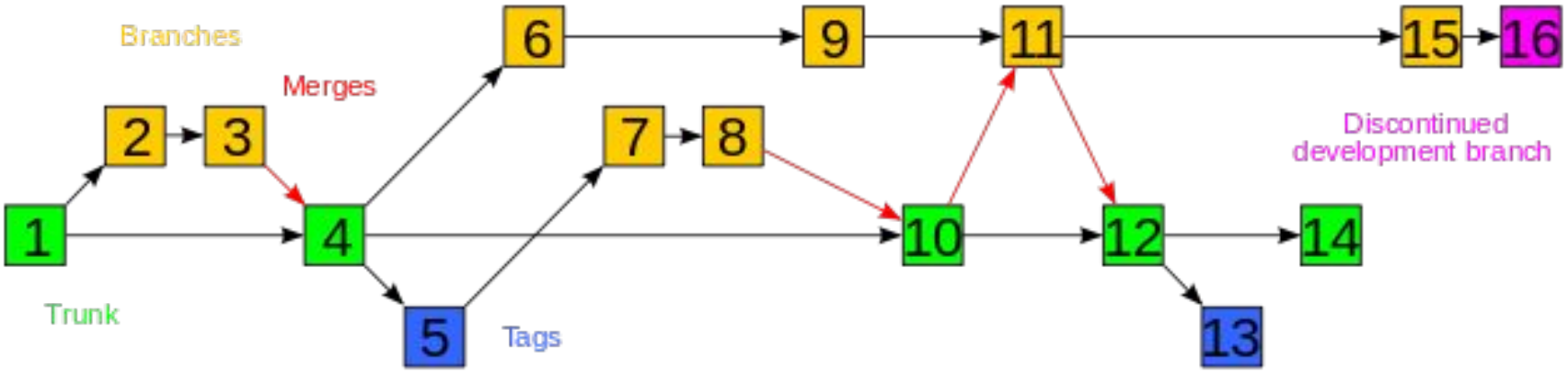
https://git-scm.com/book/en/v1/Getting-Started-About-Version-Control

Commands

- Local operations (same as previous in CVCS):
  - update/commit
  - tag/branch
  - log/di
- Remote operations (new concepts in DVCS):
  - Pull: Get all *changesets* from origin *repo*
  - Push: Send your *changesets* to origin

Branch and Tag

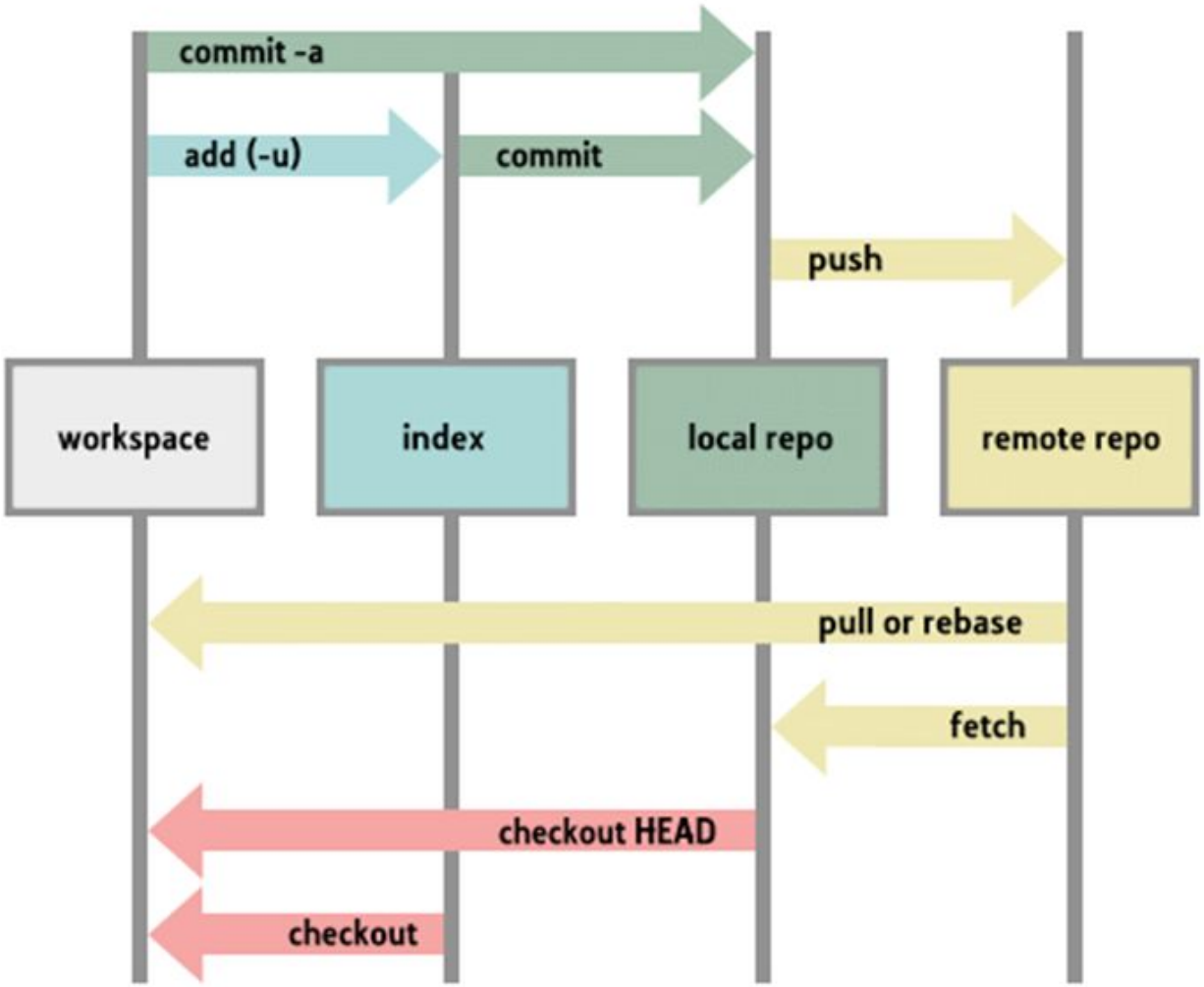
- A branch is a separate line of development.
- Tagging refers to labeling the repository at a certain point in time so that it can be easily found in the future.



Git

- Designed for code
- What git does for you:
  - Version control and source control
  - Tracks all changes submitted to the codebase
    - Revert changes
    - Create working branches without affecting stable releases
    - Tells you when there are merge conflicts
- Why git?
  - Because “everyone” is using it
  - You’ll likely use it in industry

Git Structure



Q&A

Thanks!