AGENDA

Version Control

Git intro

GitHub intro


Source Control

What is Version Control?

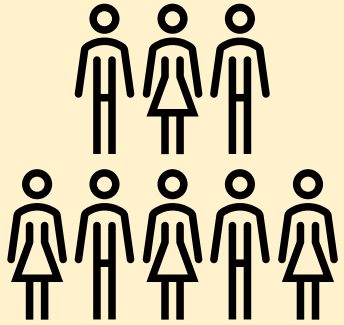
- Aka “Source Control”
- Keeps track of your creative output
- It tracks **what** is changed
- It tracks **who** made the change
- It tracks **why** changes were made

Why Version Control?

Content

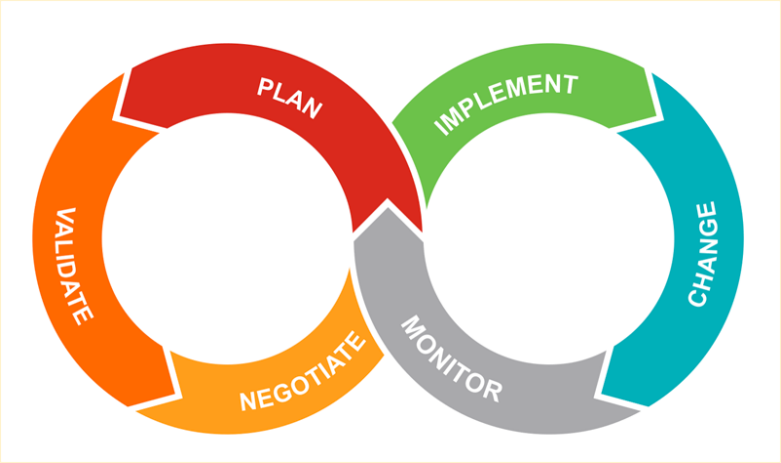
Complete history of the project is trackable and available at any time

Teams



Supports many workflows:
1- collaboration
2- team review
3- communication

Agility



1. Manages small changes
2. Easy testing (new ideas)
3. Easy fix or undo changes

Who needs it?

Who is it for?

- Developers/ designers
- Writers/ producers /Artists / composers

What do it use it for?

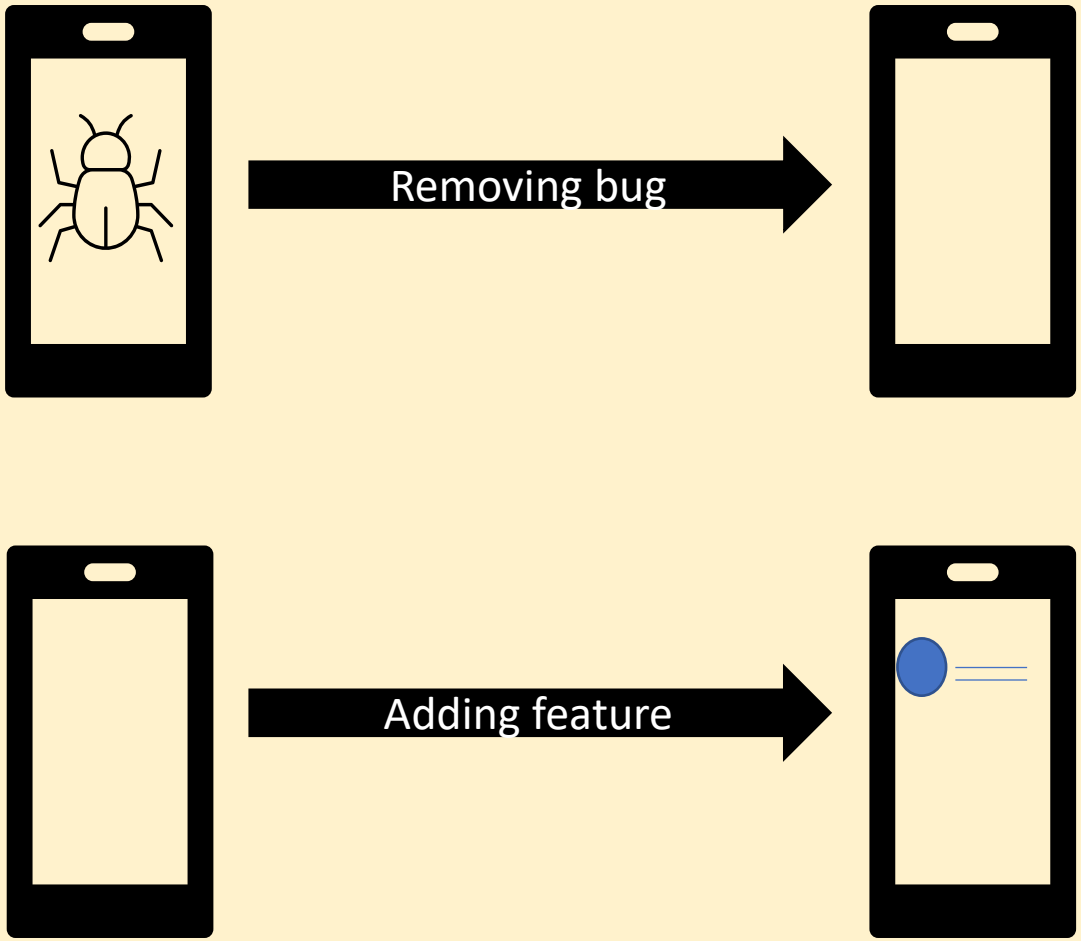
- Source code, scripts, images, icons, style sheets
- Novels, screenplays, spreadsheets, music
- Alone, or with team

Big Idea - Collaboration

1. **Team Benefits**
    - Synchronization
    - Accountability
    - Conflict Detection

2. **Social Coding** - open source projects

It will let you deliver **different versions** of the product (*Continuous Improvement)*



Vocabulary – general terminology

- **repository** - your tracked project, repo
- **main (used to be master)** - baseline repository
- **check-in** - combining your work with master
- **check-out** - updating your local work with latest files from master

AGENDA

Version Control

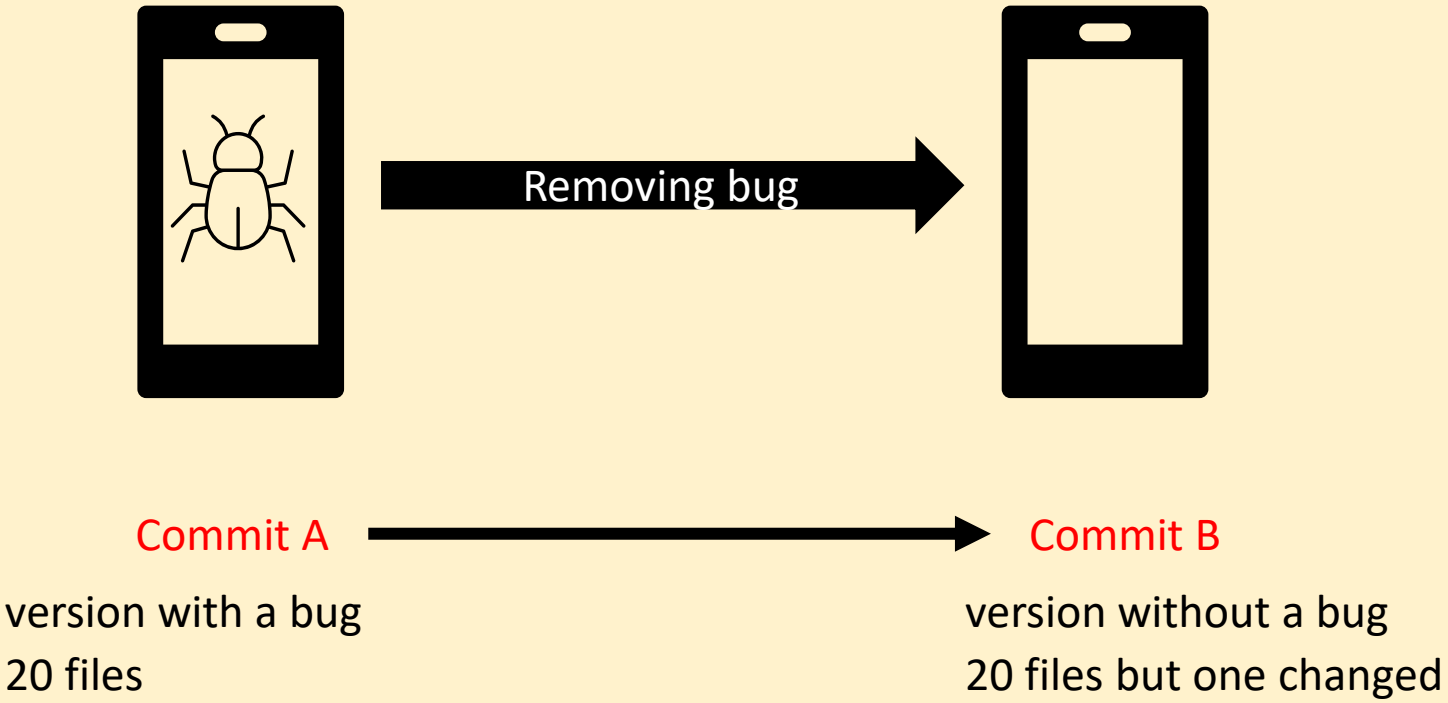
Git intro

GitHub intro

What is “git” ?

- “Distributed version control”
- Free, open source (2005, Linus Torvalds)
- Command-line tool
- Runs on all OS’s
- Efficient, simple, distributed, scalable, “light”

- **Git manages versions of the project**
- **Each version called a commit.** (even a small change)
- **Each commit is a new snapshot of the entire project.**



- **Behind the scene, git is very efficient in storing commits (versions)**
- **Each unique file is stored only once**

Commit A

version with a bug

20 files

20 files stored

Commit B

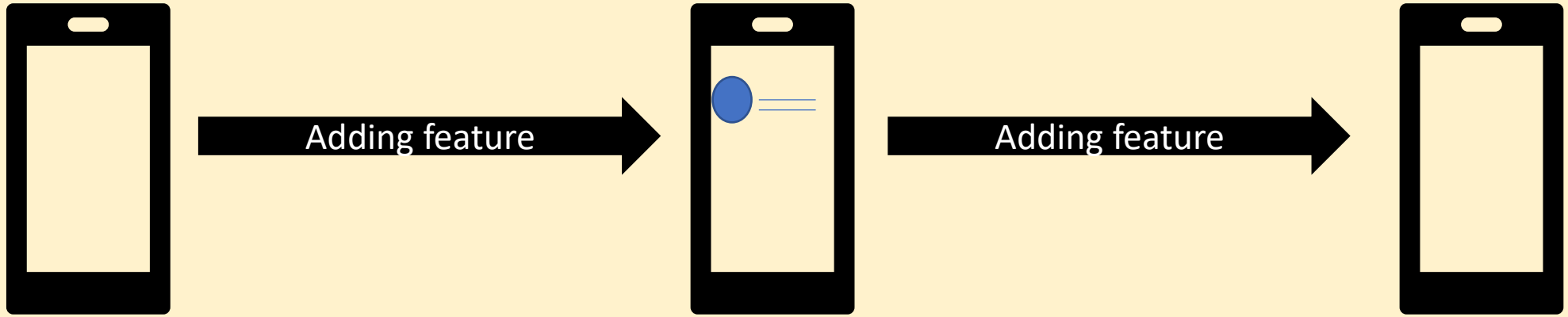
version without a bug

20 files but one changed

21 files stored

Collection of commits contain the history of the project.

- **You can review the history**
- **You can “undo” a change (travel back in time)**



Why remove a feature?

- User Testing shows that it is not useful
- Add problem you didn’t foresee
- Client didn’t like the feature
- …

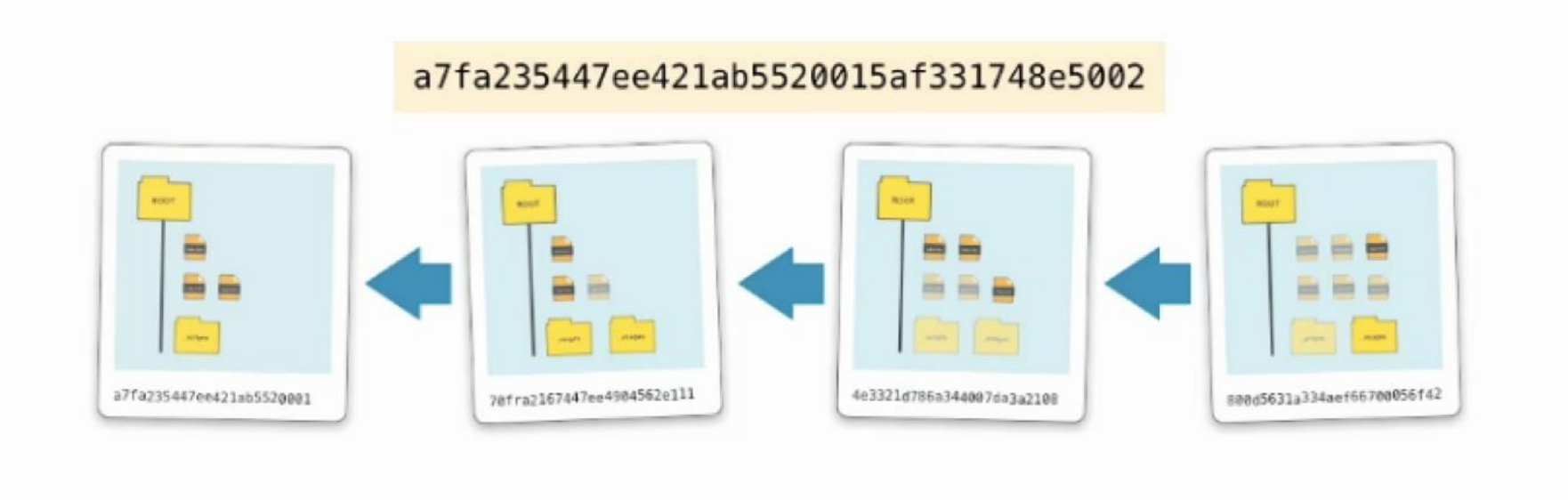
Basic commands and actions…

1. **“git clone”** - to replicate remote repo to **local** computer
   “git init” - to turn a local dir into a **local** repo
2. **“git add”** - to add (stage) files from your **local** dir to the **stage/index**
3. **“git commit”** - to move what’s on **stage/index** to **local** repo
4. **“git push”** - to replicate local repo to **remote** repo

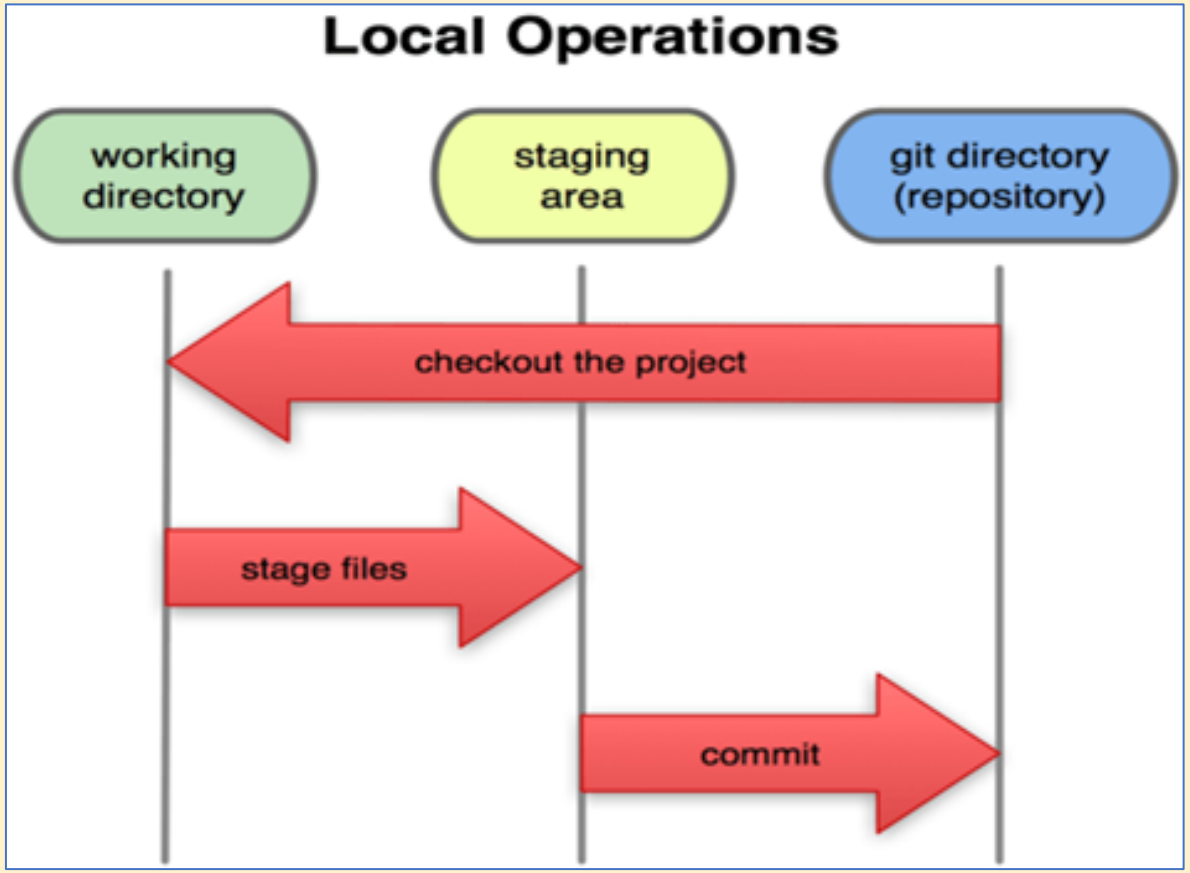
Memorize this pattern:

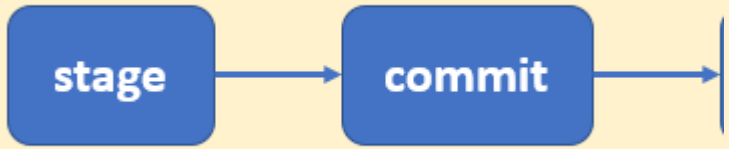
To “commit” (a verb and a noun)

- 40 character string hash reference for each commit.
- Git reduces it to 7 characters in the log
- Each commit references an older commit.



Key “git” concepts, visually...





Stage/ index

Working directory

Repo

Key “git” concepts...

The files (in your working set) have 3 possible status:

1. **Modified/Untracked** (files are modified/new)
2. **Staged** (modified files are set aside)
3. **Committed** (staged files are safely stored into repo)

Another way to look at it: files belong to 3 virtual spaces:

1. **Working Directory** (untracked files, or modified files)
2. **Stage (Index)** (modified files are set aside here)
3. **Repository** (“the” project)

All on Local Machine

AGENDA

Version Control

Git intro

GitHub intro

What is “GitHub”

- “Web-based git repo hosting service”
- (Git is command line tool)
- Github is remote server, graphical, and ...
- Flagship features: fork, merge, pull

To let others see your work ...

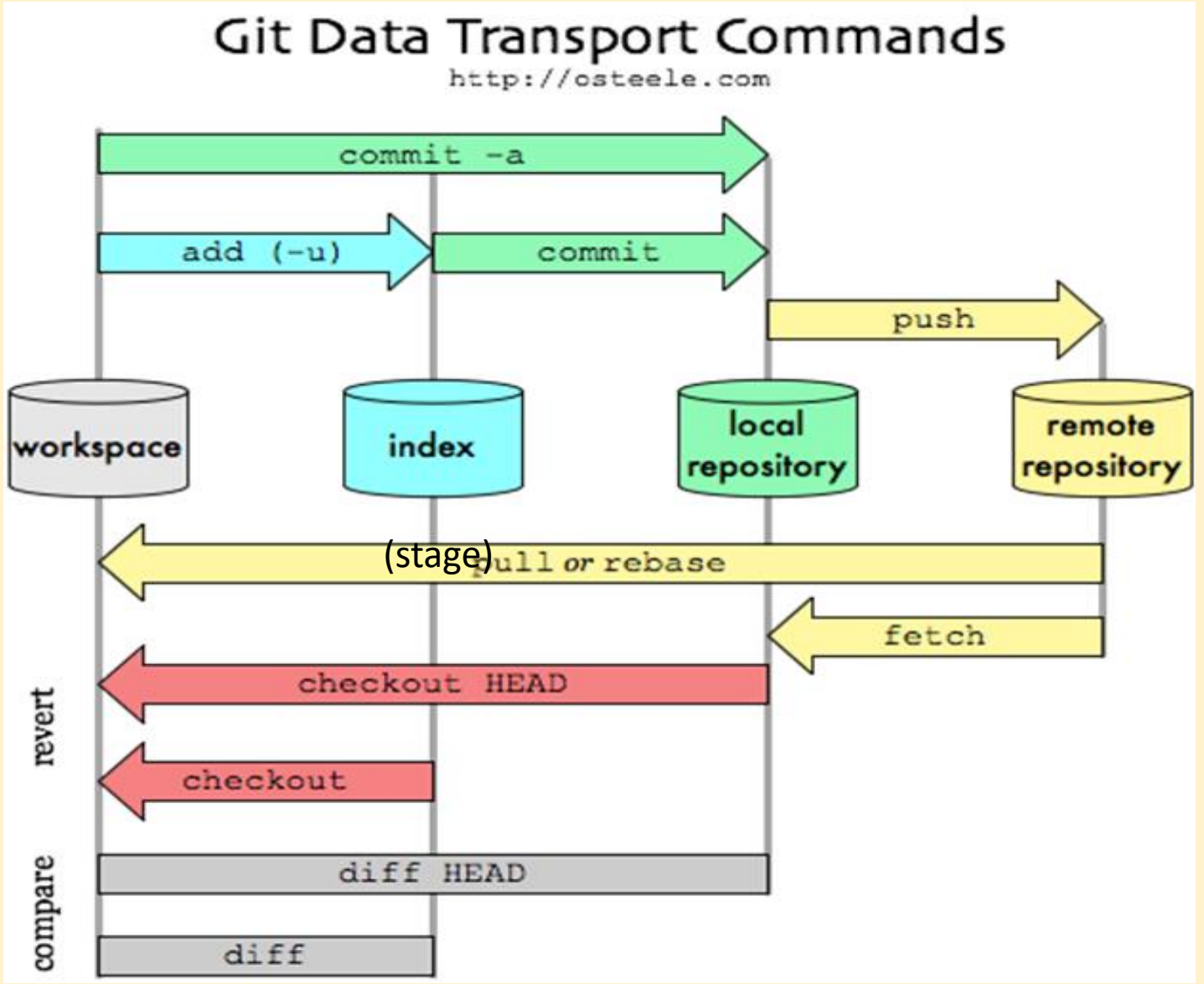
- **git push**
  - Copy your local, committed, repo to a remote location.
  - Can push either a whole project, or one branch
  - You could be “unable to push”. Solution:
    - Pull (fetch and merge), Fix conflict,
    - Push again

Synching local machine with remote

What is “.gitignore”

- This is a “hidden” text file at your top level
- List of files (or file formats) you don’t want to include when you “push” to remote.
- Some machines or programming languages have intermediate or local config files others don’t need to see.
- Examples: .DS_Store, *.class, *.obj

Recap Big Picture



stage → commit → push

AGENDA

Demo:

Getting started locally

Demo

1. Follow the demo
2. Submit your final work