GIT - 1 of 4

SIMANTA MITRA

Acknowledgments:
Many of the illustrations and information is from various Web sources.
Key ones:
https://juristr.com/blog/2013/04/git-explained/
http://rogerdudler.github.io/git-guide/
http://marklodato.github.io/visual-git-guide/index-en.html
https://onlywei.github.io/explain-git-with-d3/

Overview of GIT topics

- In GIT-1, we present basic GIT concepts.
- In GIT-2, we focus on local repository operations.
- In GIT-3, we focus on branching operations.
- In GIT-4, we focus on remote repository and operations.

Learning Objectives

- This is a basic introduction to GIT. GIT is a version control mechanism.
- Motivation: Why is a tool like GIT a necessity for Software Development?
- Basic Concepts
  - Local and remote repositories
  - Working Directory, Stage/Index, History
  - Commit or snapshot
  - tracked, untracked files
  - .gitignore file
- A few git operations
  - init
  - add, commit (including modeling)
  - status, log, ls-files, ls-tree

Collaboration, History

MOTIVATION

Source Control: Motivation

Has this ever happened to you?

- It is late at night and your code does not work perfectly.
- You have a great idea about how to fix it!
- You make massive changes…

- Many hours later…. you wish you could get your old c… At least that worked! (Need to have backed up)

- One way: main.c.bak, main.c.bak1 etc

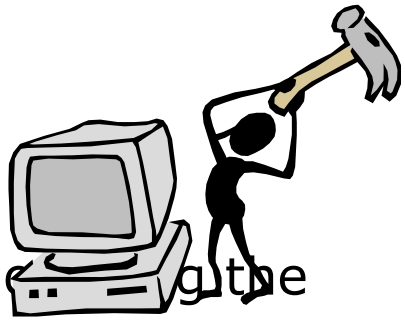
Source Control: Motivation

- Suppose there is a server which has files A and B.
  - Jack and Jill copies A and B to their PCs
  - Jack changes A and saves back to server
  - Jill changes B and A and saves on server

  - Jack's work is lost because of multiple people accessing and modifying the same files!

Source Control: Motivation

- Suppose there is a server which has files A and B.
  - Jack and Jill copies file to their machines.
  - Jack modifies A and saves Aa
  - Jill modifies B and saves Bb
  - Jack modifies B and saves Ba
  - Jill modifies A and saves Ab
  - Jack copies Ab and Ba to his machine
  - Jill copies Ab and Ba to his machine
  -
  - Original copies of A and B is lost.
  - changes made by Jack to A is lost.
  - changes made by Jill to B is lost.



  - work is lost because of multiple people accessing and modifying the same files!

Motivation

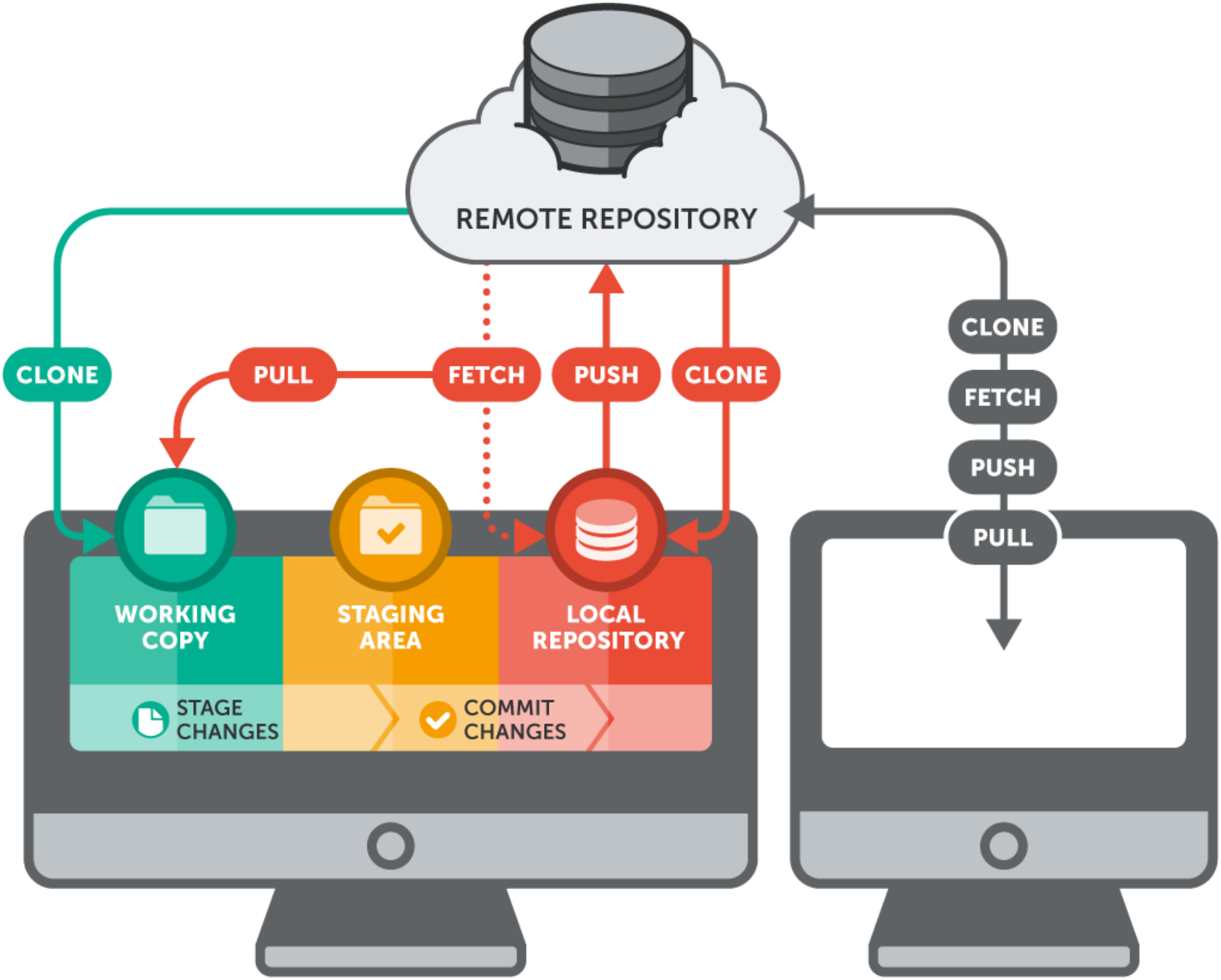
1. maintaining backups
   - What/When/who changes were made?
   - Revert to pervious versions
   - What code was present in release 2.1

2. concurrent Development

Do note that it is useful for even a single developer to use source control (why?)

Local/remote, wd, stage, history, tracked, untracked, commit, .gitignore

BASIC CONCEPTS


REMOTE REPOSITORY
CLONE
PULL
FETCH
PUSH
CLONE
CLONE
FETCH
PUSH
PULL
WORKING COPY
STAGING AREA
LOCAL REPOSITORY
STAGE CHANGES
COMMIT CHANGES
DEVELOPER A
DEVELOPER B

Local Repository

Actually .git directory also holds stage info

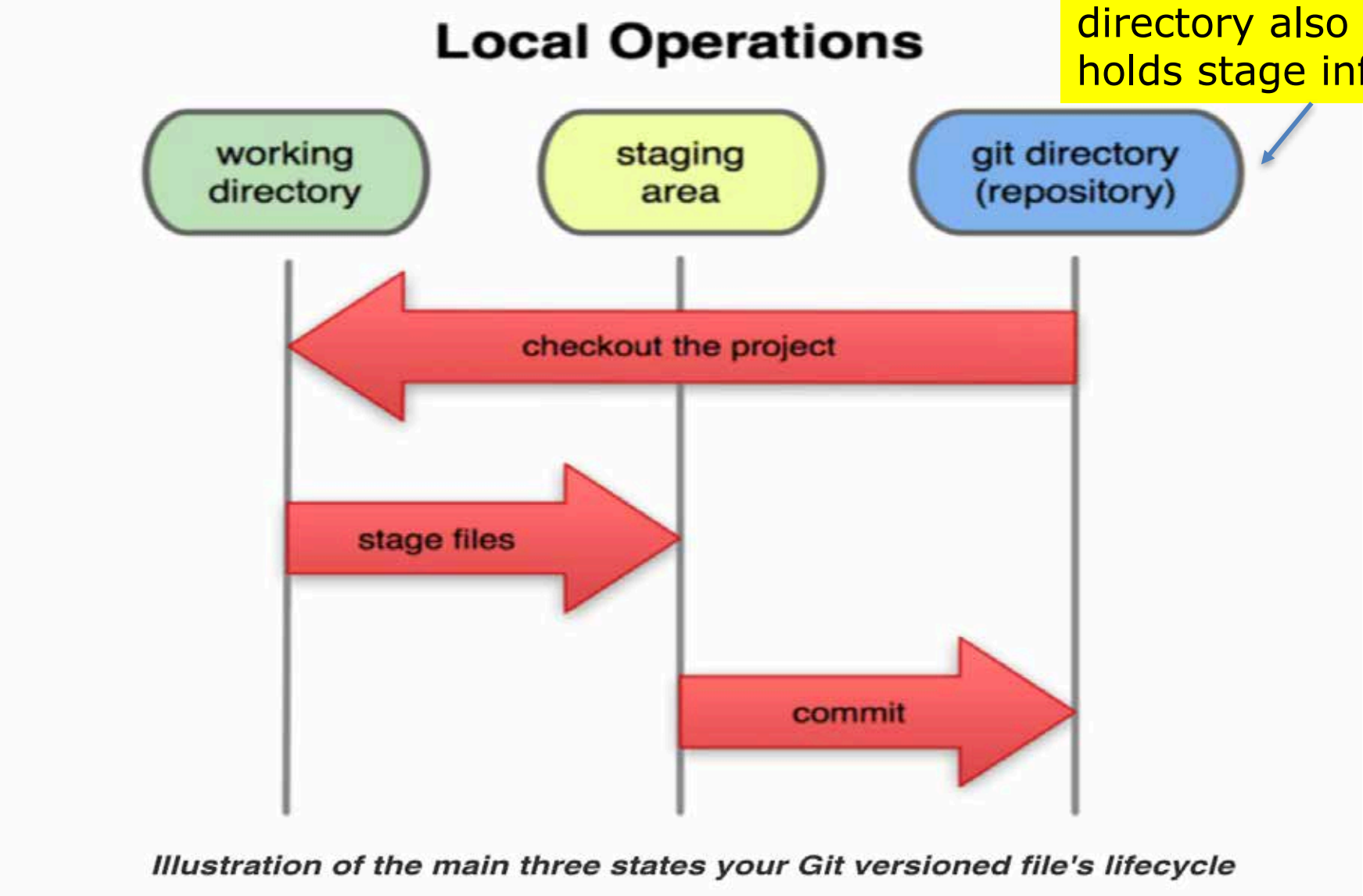
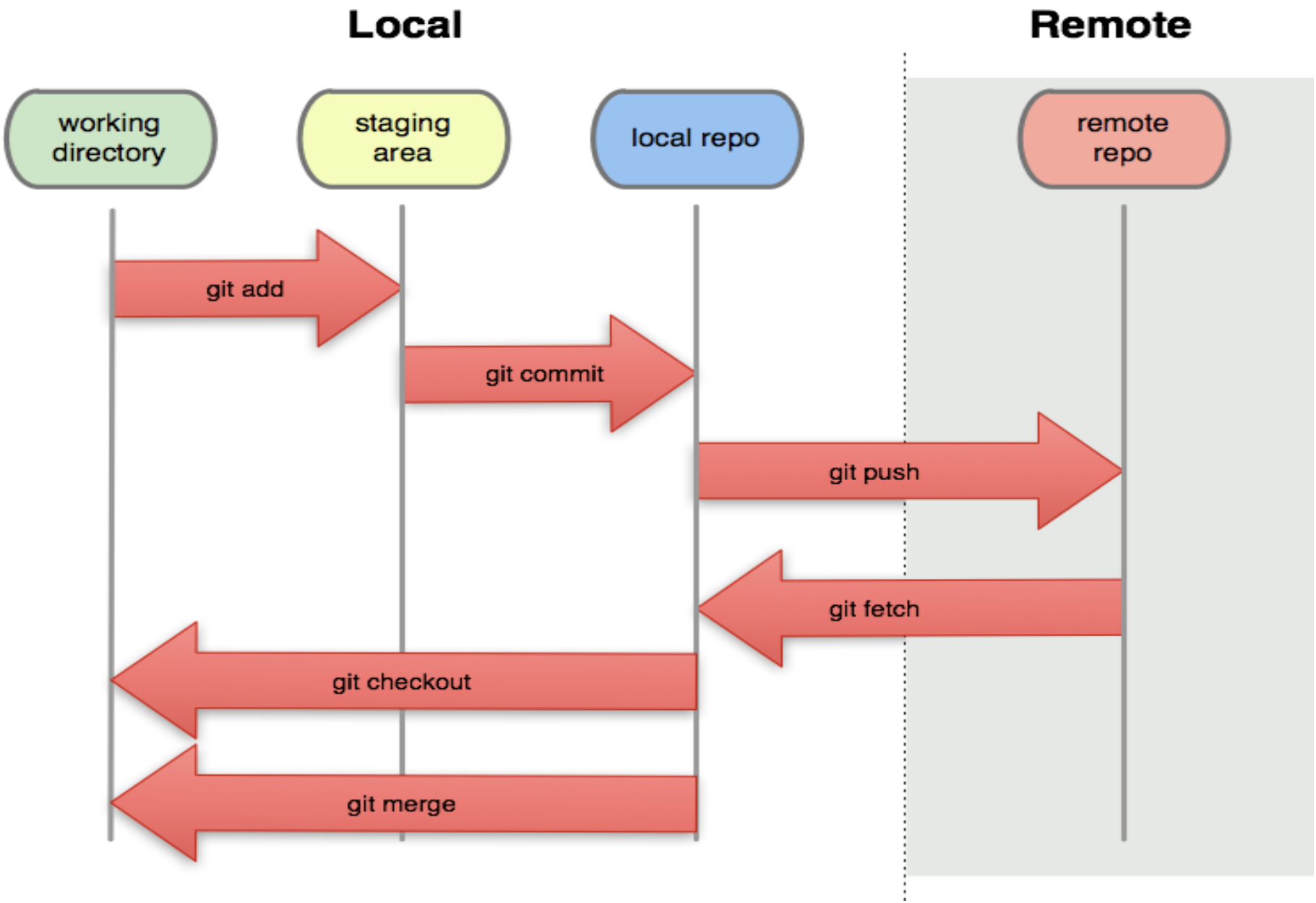
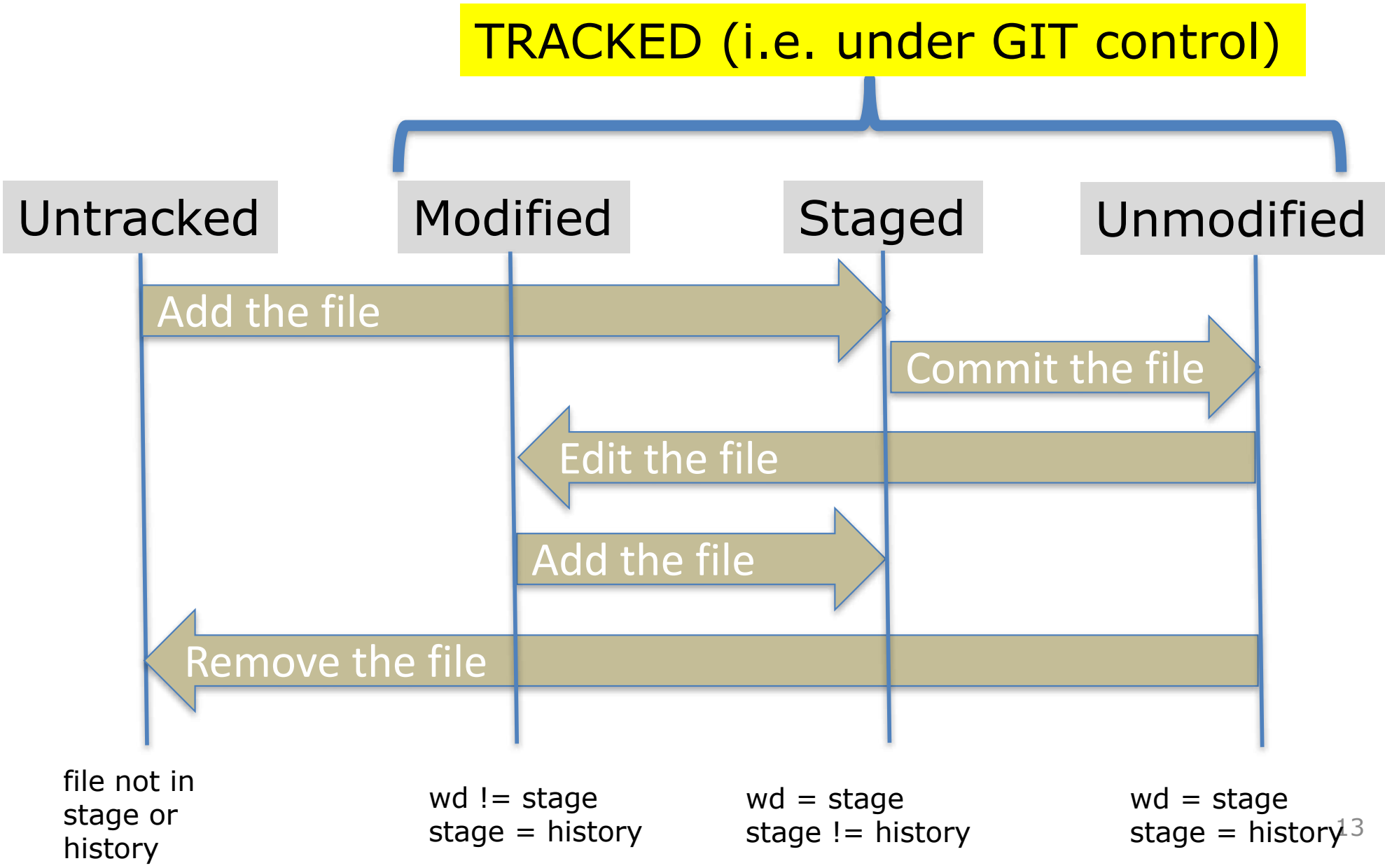


Illustration of the main three states your Git versioned file's lifecycle

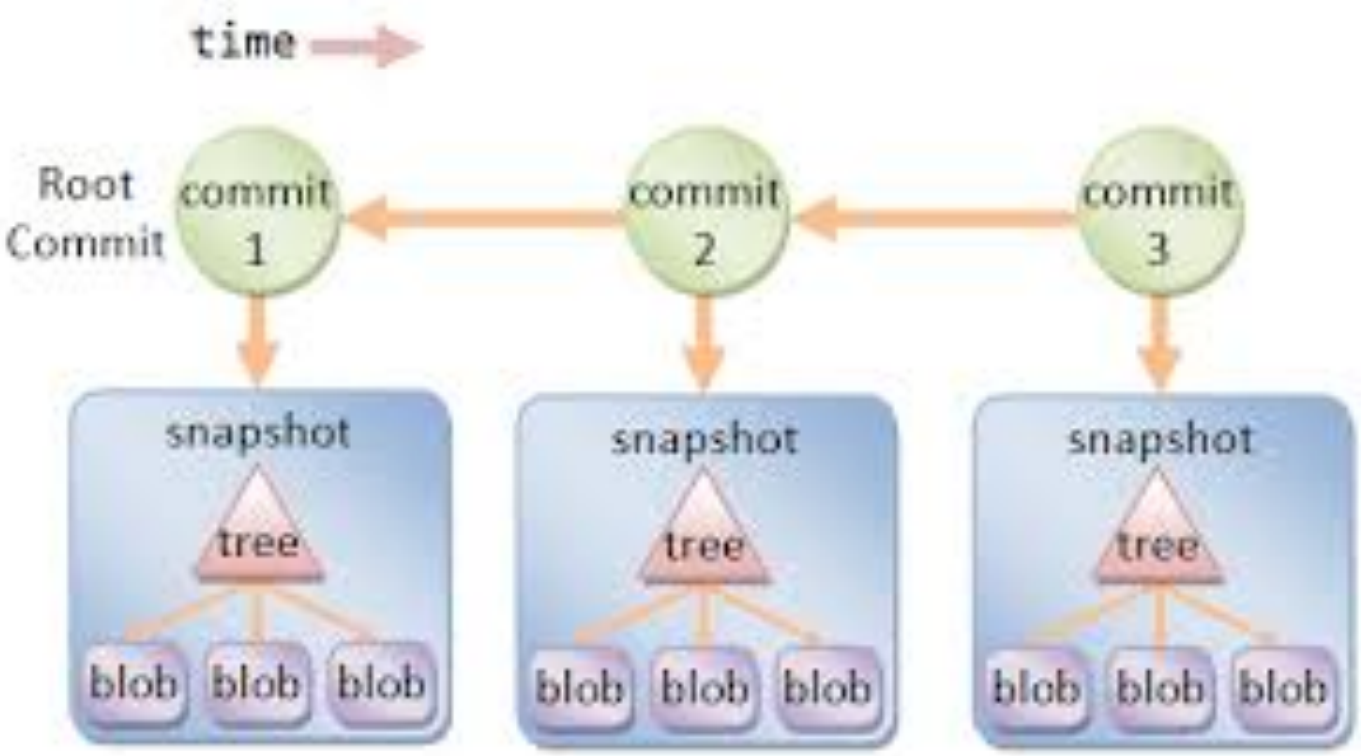
Local & Remote Repository



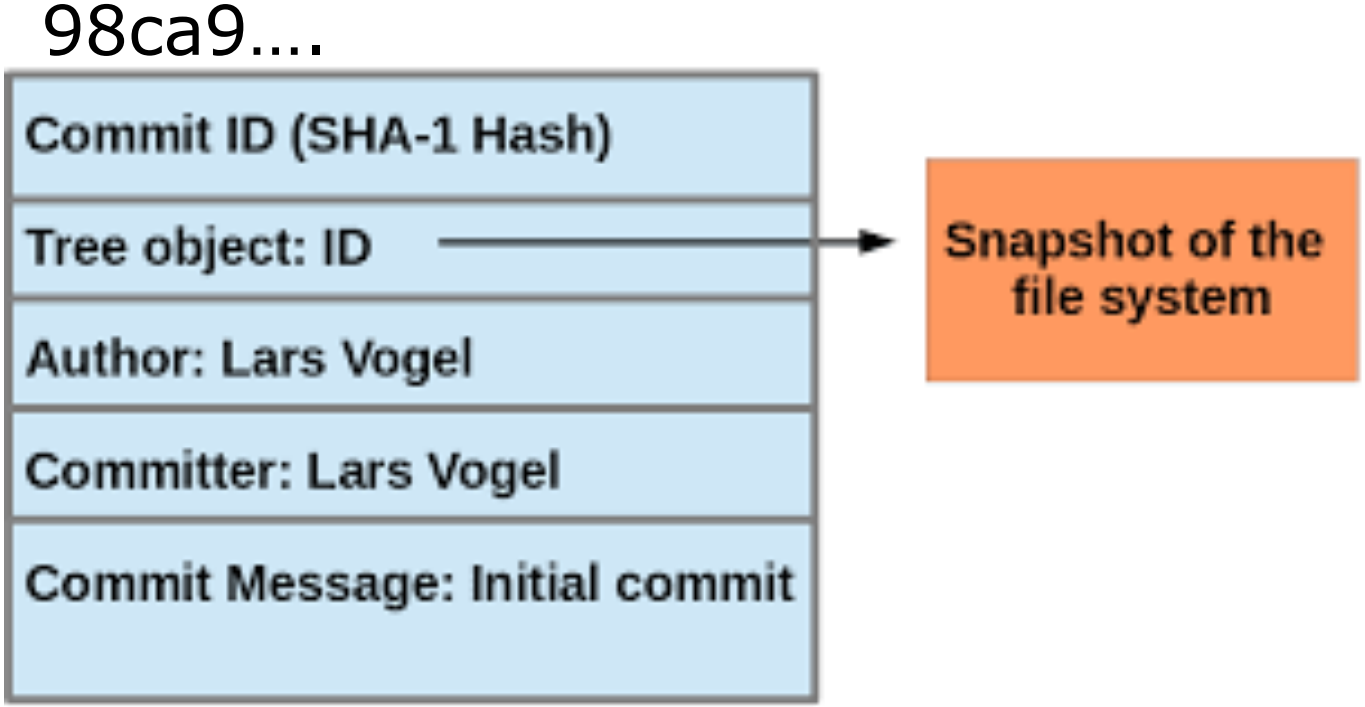
State of files and directories



Commits in history



A Commit object



Terms

- Remote repository
- Local repository
- Working directory
- Stage
- Index
- Local History (snapshots)
- Commit object
- ID of commit object

init, status, add, ls-files, commit, log, ls-tree

A FEW OPERATIONS

git init

git status

git add

- **.gitignore (***** VERY VERY VERY IMPORTANT)**

git ls-files --stage

git commit

git log

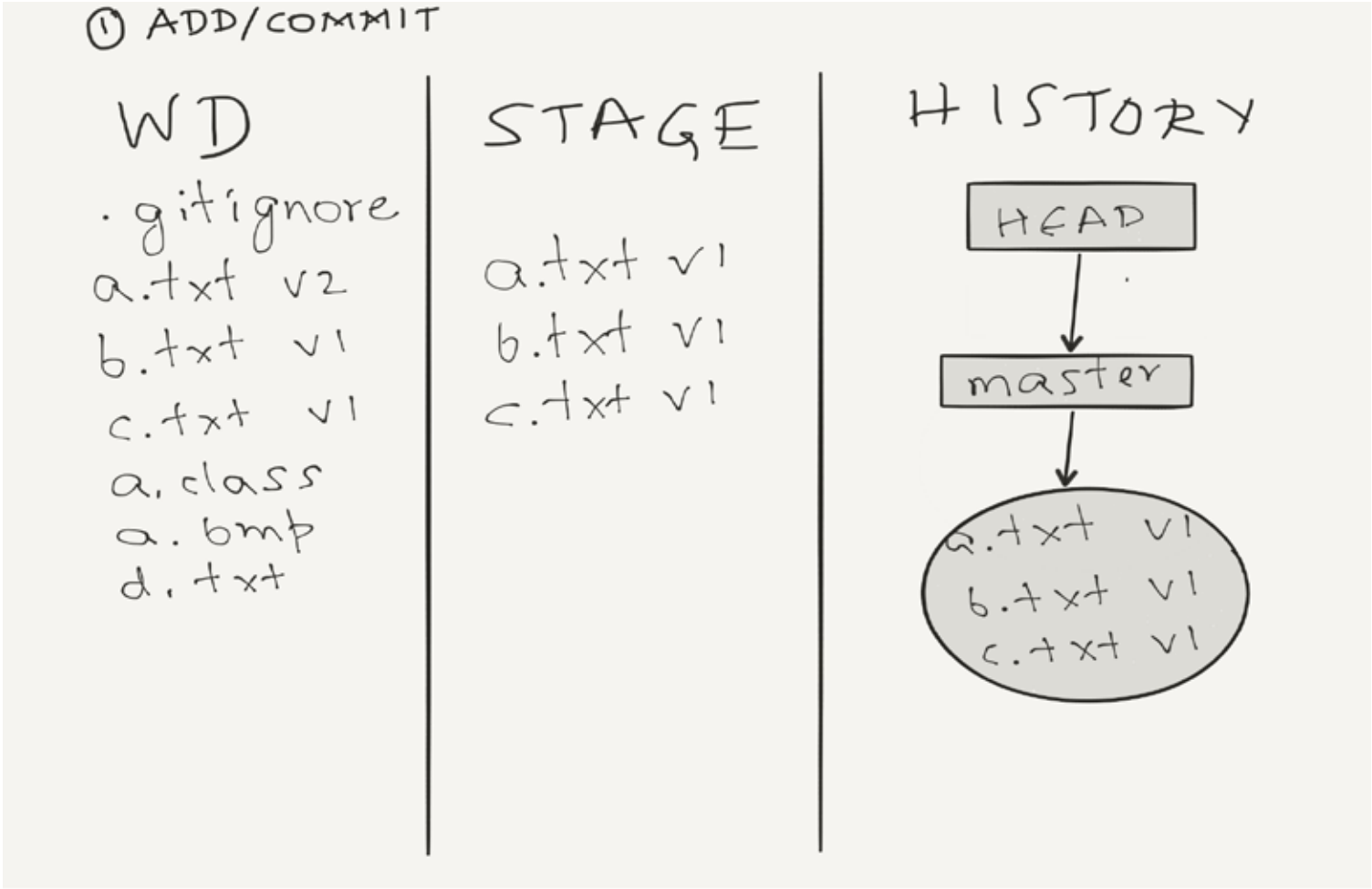
git ls-tree

SELF CHECKS

Self Check-1

- What are the two main reasons that developers need to use a tool like GIT?
- Would a tool like GIT be useful for a single developer? Explain.
- What are two reasons that you need to push your work often?

Self Check-2



- Assume that .gitignore has *.class

What happens after a git add . (draw the result)

Self Check-3

- What happens on a "**git commit**" command made after the previous git add command. Draw the end result.

Self Check-4

- What is a commit object? Describe in a few sentences.

THE END!