Git & GitLab

A brief introduction

Outline

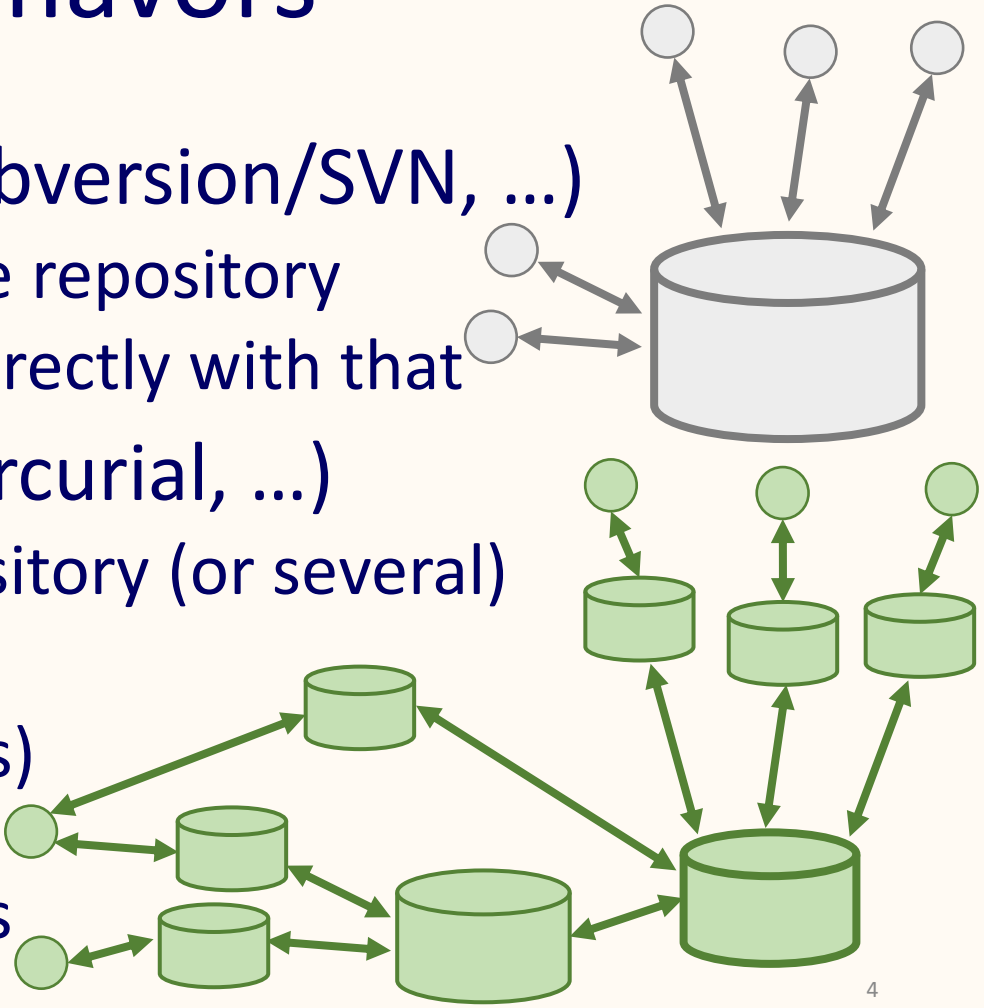
- Basic terms/concepts
- Common usage
- First lab

Version control systems (VCS)

- Manage changes to documents
  - Especially team software development
- Major features:
  - Long-term change history
    - What files changed when, & who changed them
    - Allows you to restore specific versions later (if needed)
  - Branching/merging
    - Allows groups to work separately, then recombine
  - Integration with bug tracking, other tools

Two major VCS flavors

- Centralized (CVS, Subversion/SVN, …)
  - A single authoritative repository
  - Everyone interacts directly with that
- Distributed (Git, Mercurial, …)
  - Everyone has a repository (or several)
  - Work is done locally
  - Change-sets (patches) communicated between repositories



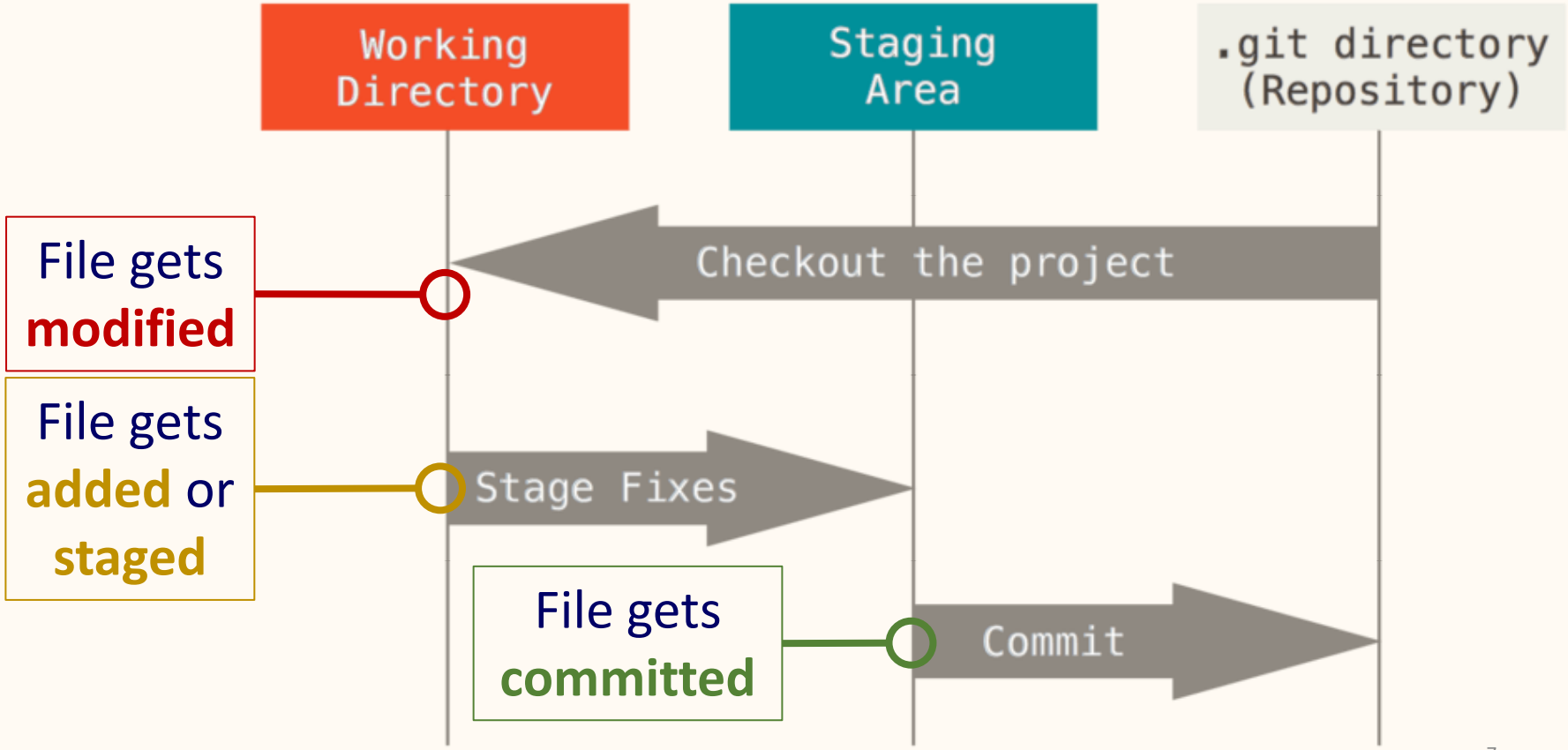
Git & GitLab

- Git:
  - Open-source, distributed version control system
  - Widely used (e.g. GitHub, BitBucket, …)
  - Most IDE's can interact with Git repositories
- GitLab:
  - Locally hosted DevOps platform that includes Git
    - We have it installed at **`gitlab.cs.unh.edu`**

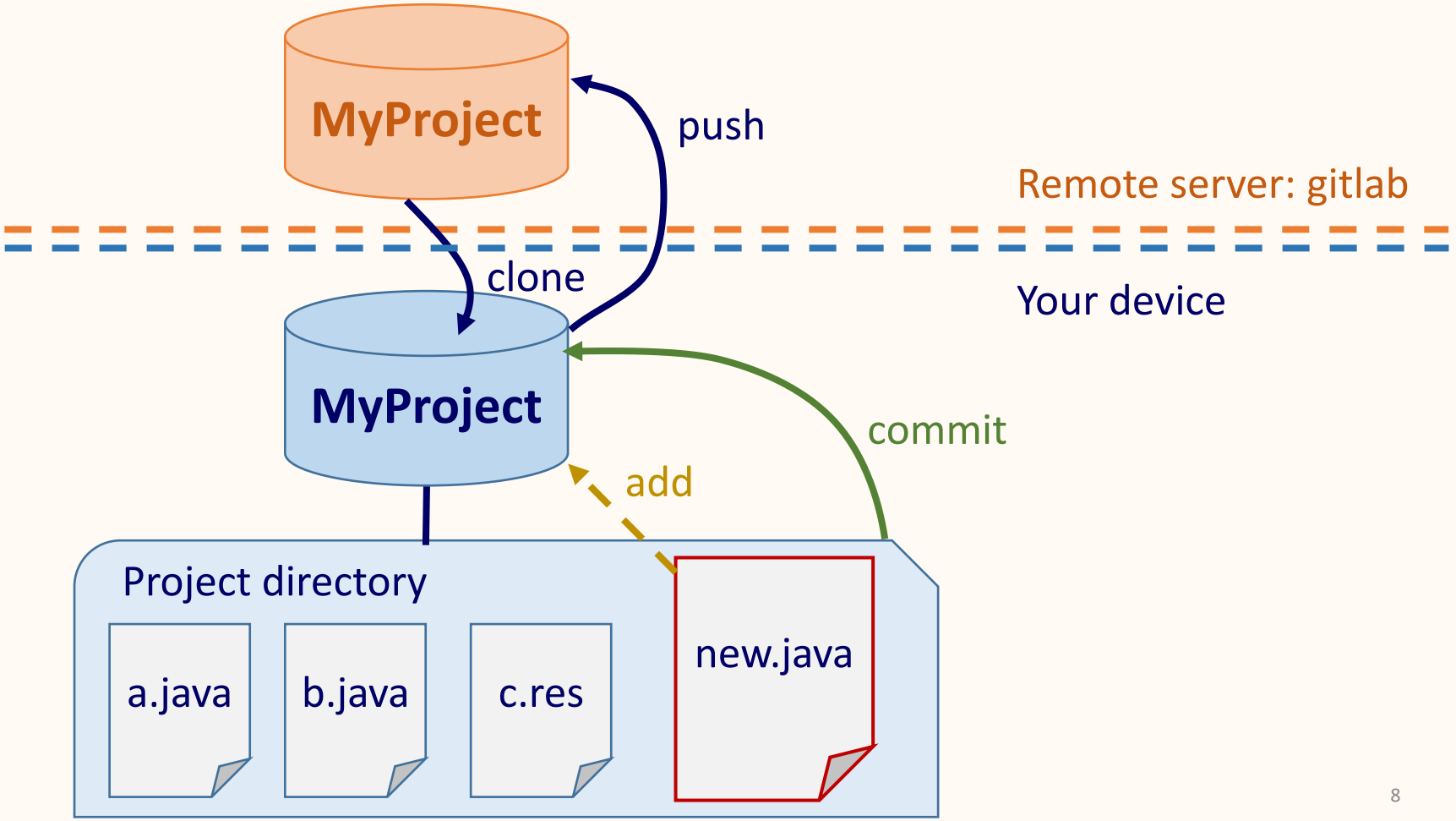
How Git works locally

- Key concept: 3 states for your files
  - **Modified**—file is changed but Git doesn't know
  - **Staged**—Git has marked files as modified file, and it will go into the next commit snapshot
  - **Committed**—changes to the file are safely stored in the local Git database
- Leads us to 3 main sections of a Git project:
  - Working directory tree, staging area, Git directory

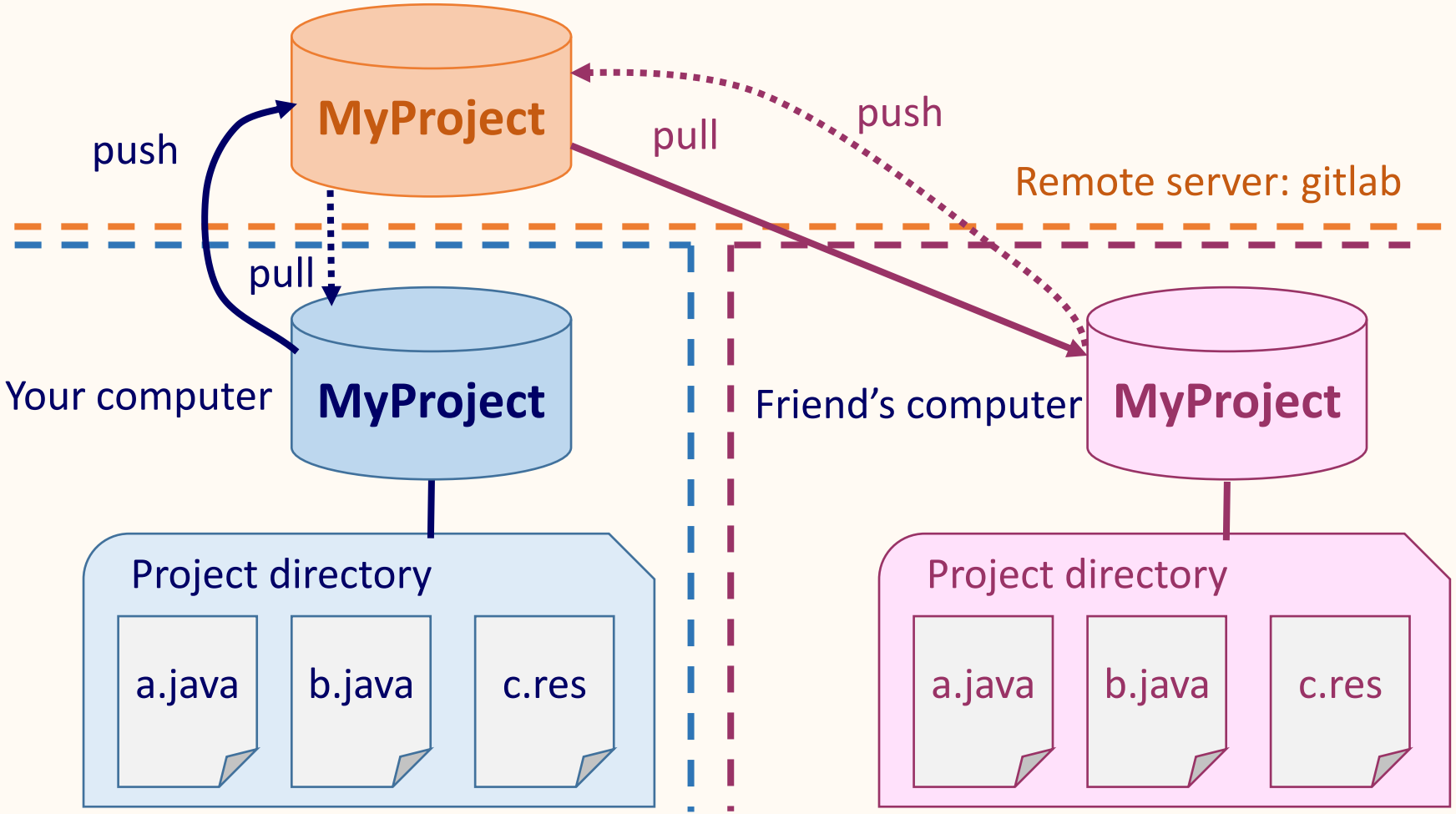
The 3 Sections and 3 States



Basic usage/commands



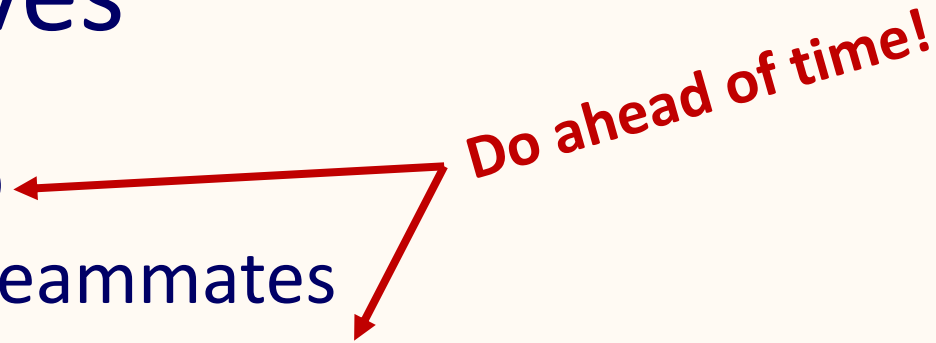
Working together (without branches)



Experiment!

- It’s hard to mess up your repository or kill your code
- …unless you forget to commit.
- Always commit your code first.
  Often.
  Always.

First lab objectives



(05) Environment setup
(10) Get to know your teammates
(05) Discord (https://csonline.cs.unh.edu/login)
(10) Terminal basics
(10) Setup GitLab
(40) Git - basics

[link to lab]