http://flic.kr/p/6oP7x7

Version Control with Git

Version Control

Why track/manage different versions of code?

Version Control

Why track/manage different versions of code?

- Backup: Undo or refer to old stuff
- Branch: Maintain old release while working on new
- Collaborate: Work in parallel with teammates

Version Control Systems (VCSs)

- Help you track/manage/distribute revisions
- Standard in modern development
- Examples:
  - Revision Control System (RCS)
  - Concurrent Versions System (CVS)
  - Subversion (SVN)
  - Git

older

newer

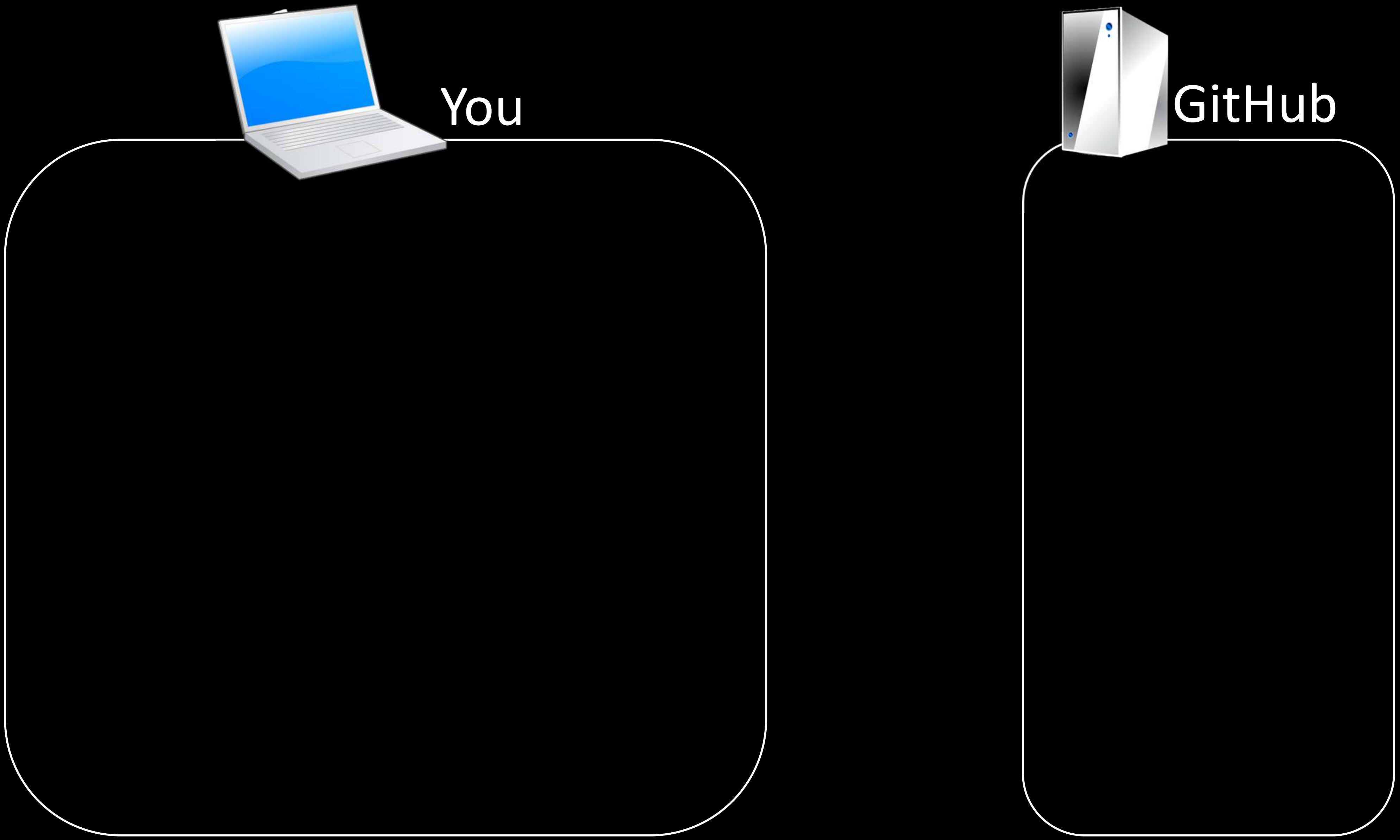
Our focus

Version Control Hosting Services

- Enable sharing version control repos
- Internet/Web based
- Examples:
  - SourceForge
  - Bitbucket
  - GitLab
  - GitHub

Our focus

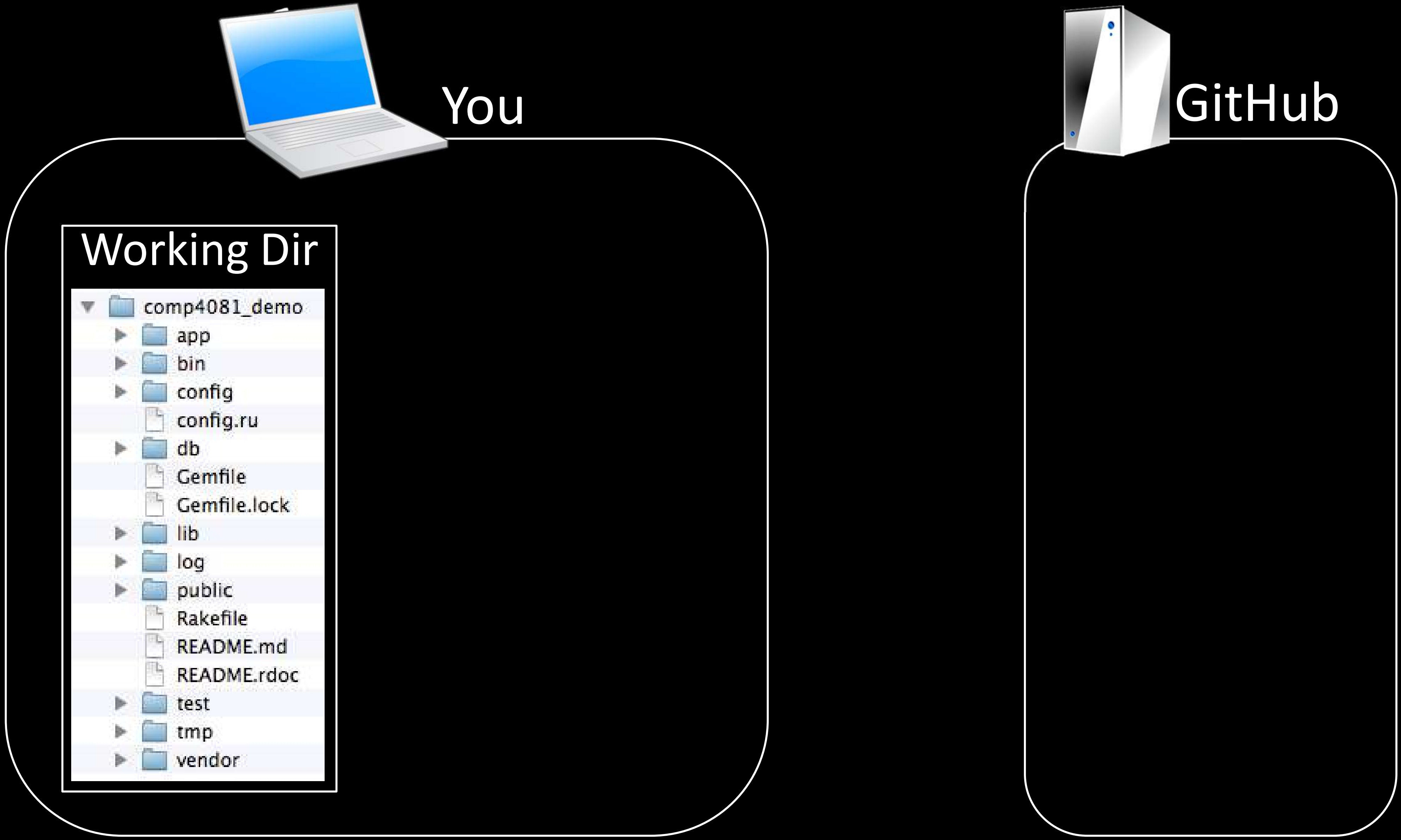
GitHub-User Perspective



You

GitHub

GitHub-User Perspective



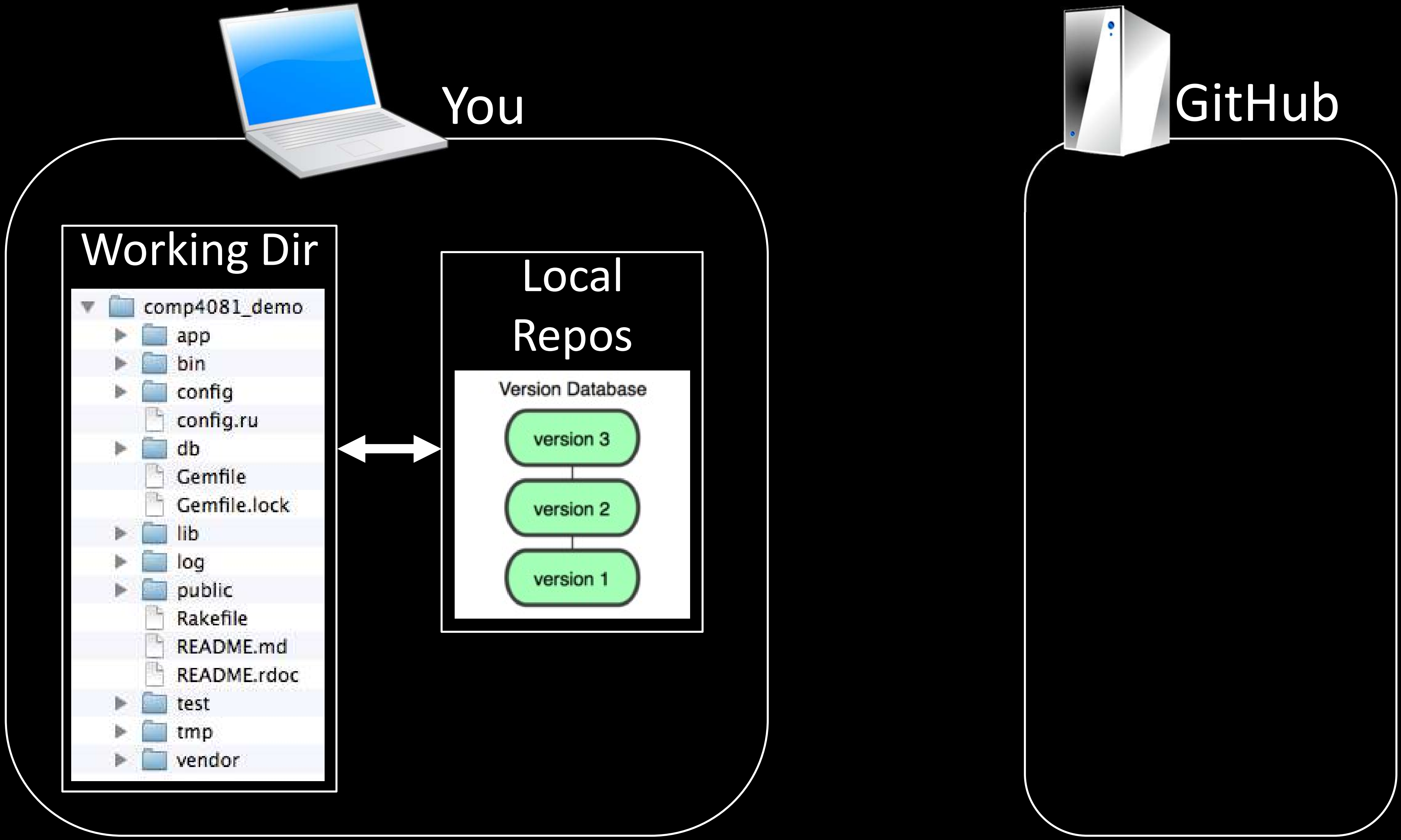
You

GitHub

Working Dir

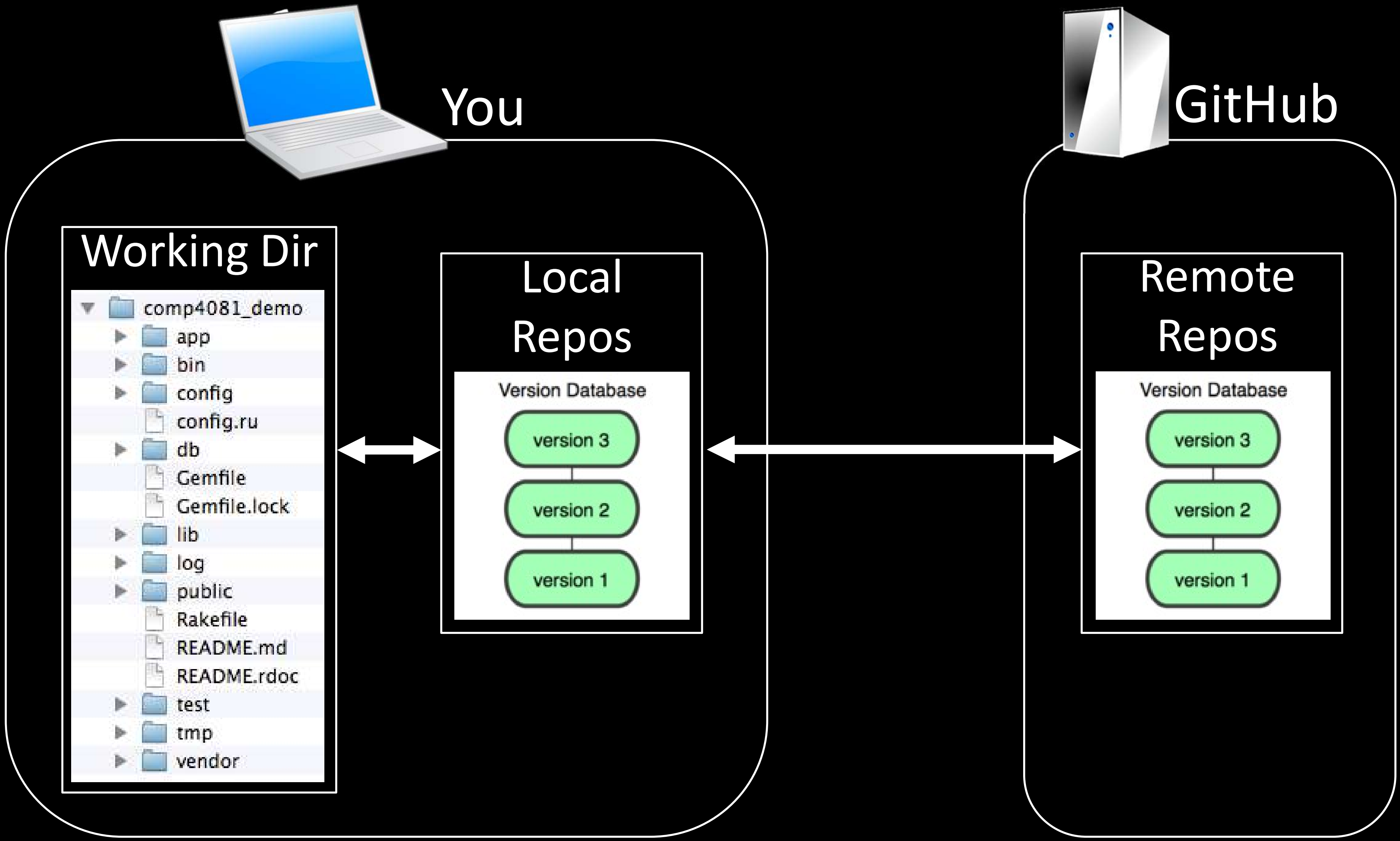
```
▼ comp4081_demo
    ▶ app
    ▶ bin
    ▶ config
      config.ru
    ▶ db
      Gemfile
      Gemfile.lock
    ▶ lib
    ▶ log
    ▶ public
      Rakefile
      README.md
      README.rdoc
    ▶ test
    ▶ tmp
    ▶ vendor
```

GitHub-User Perspective



comp4081_demo
- app
- bin
- config
- config.ru
- db
- Gemfile
- Gemfile.lock
- lib
- log
- public
- Rakefile
- README.md
- README.rdoc
- test
- tmp
- vendor

GitHub-User Perspective



You

GitHub

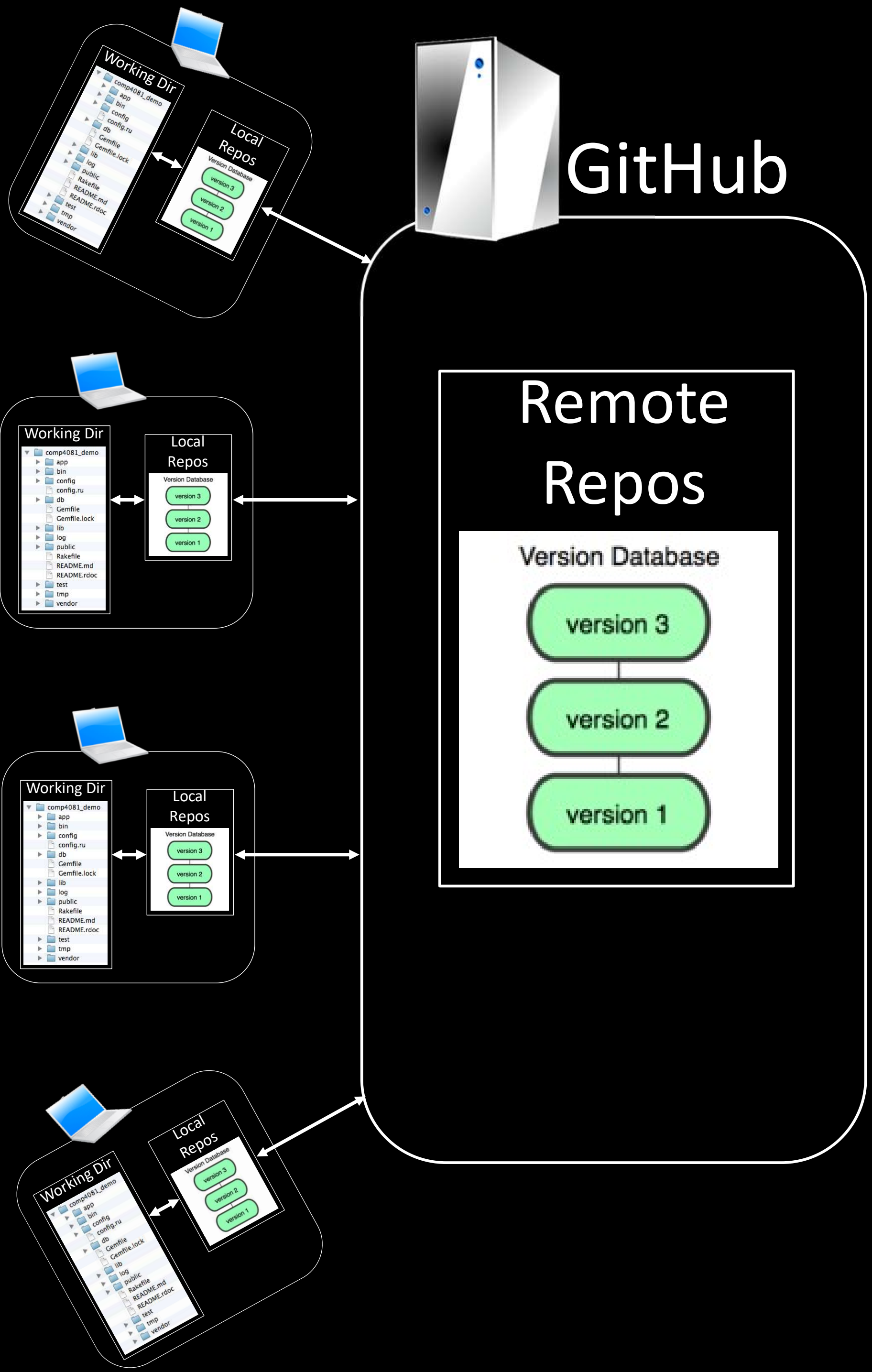
Working Dir

- comp4081_demo
  - app
  - bin
  - config
  - config.ru
  - db
  - Gemfile
  - Gemfile.lock
  - lib
  - log
  - public
  - Rakefile
  - README.md
  - README.rdoc
  - test
  - tmp
  - vendor

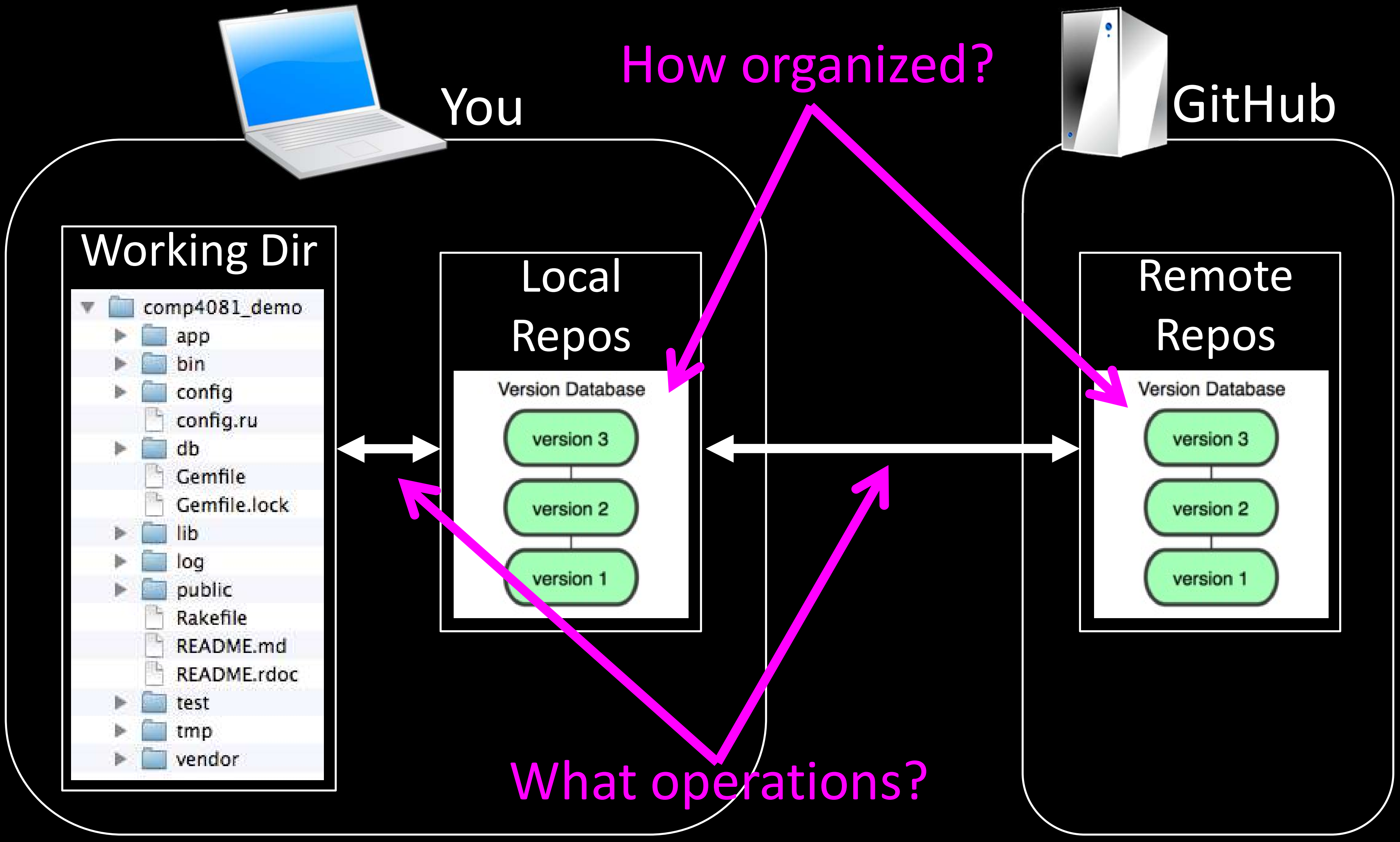
Local Repos

Remote Repos

Using GitHub to Collaborate



Questions to answer



You

GitHub

How organized?

Working Dir

comp4081_demo
- app
- bin
- config
- config.ru
- db
- Gemfile
- Gemfile.lock
- lib
- log
- public
- Rakefile
- README.md
- README.rdoc
- test
- tmp
- vendor

Local Repos

Version Database

version 3

version 2

version 1

Remote Repos

Version Database

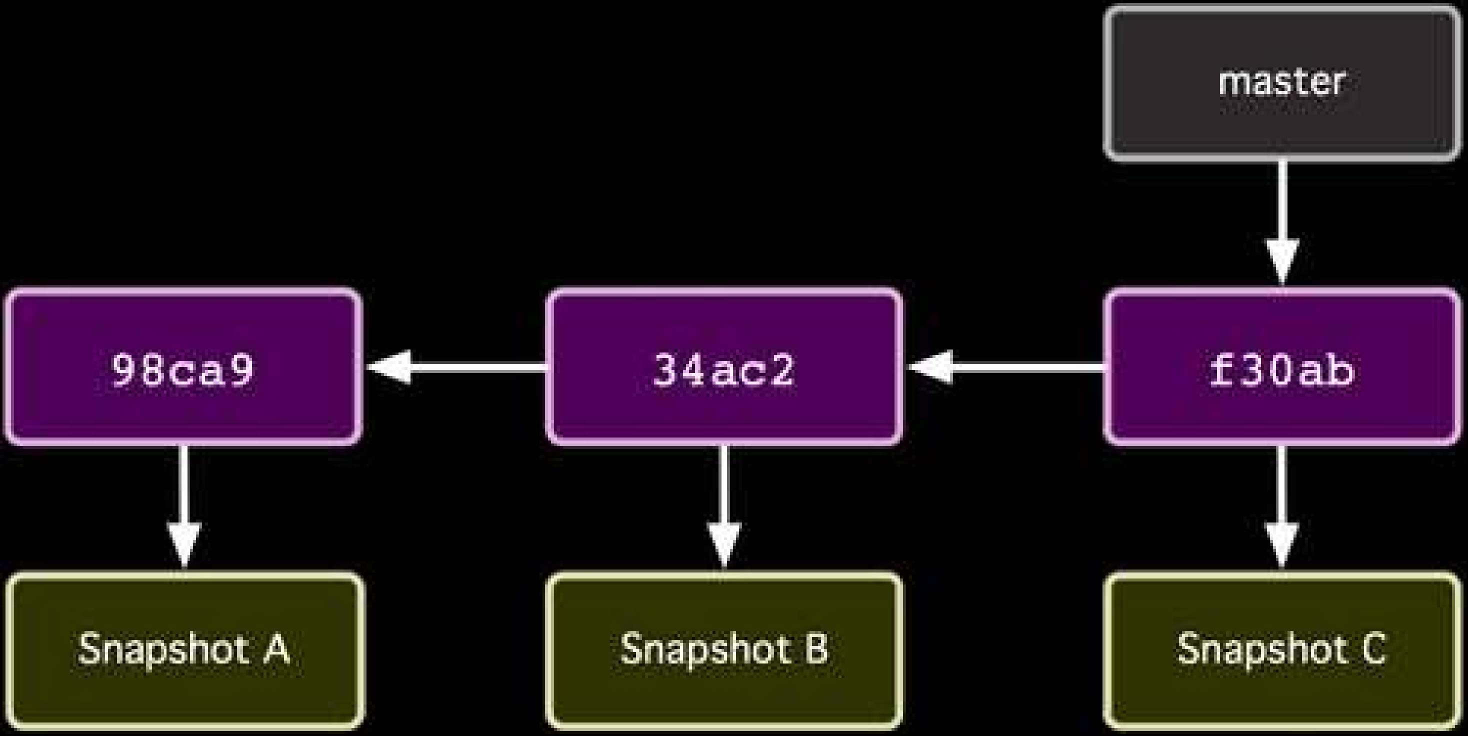
version 3

version 2

version 1

What operations?

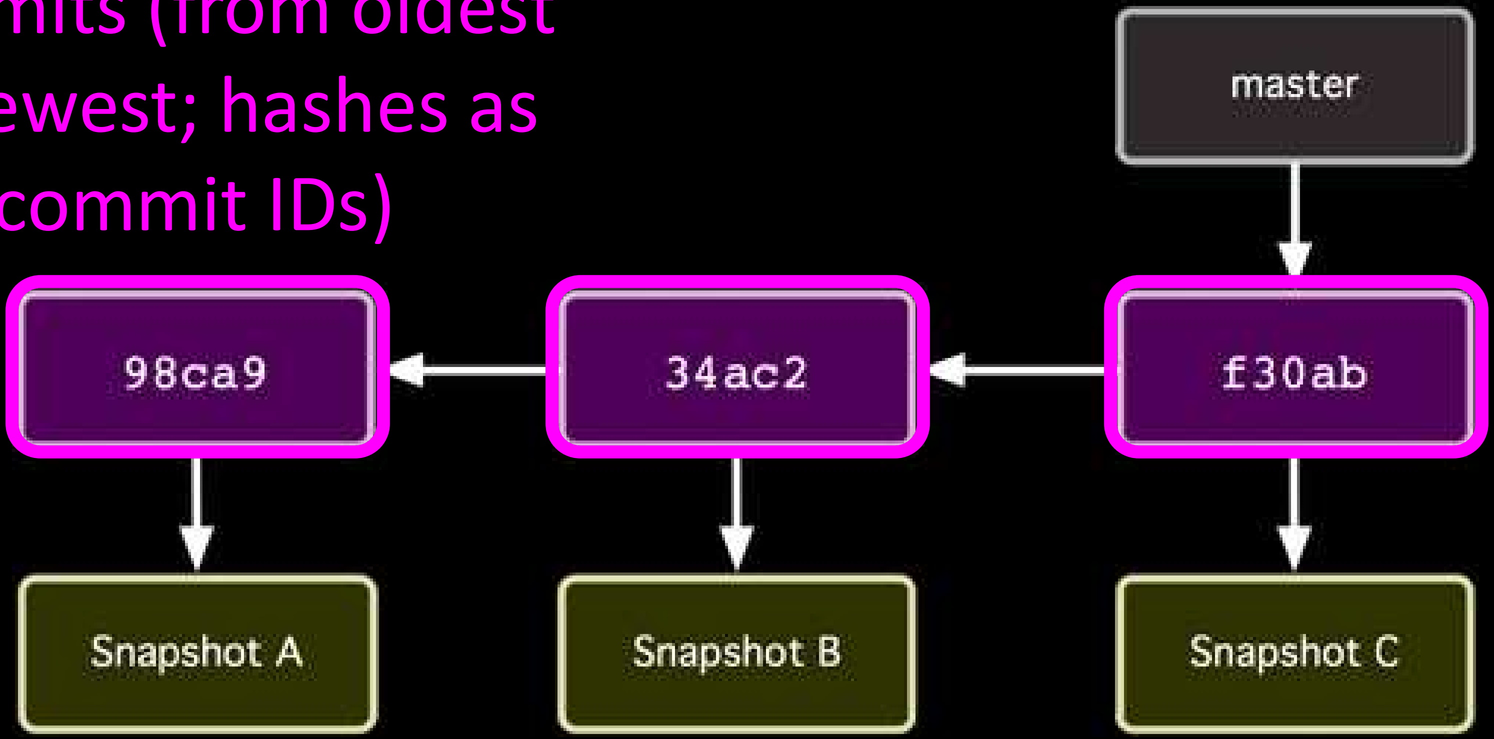
Repo Organization



http://git-scm.com/book/

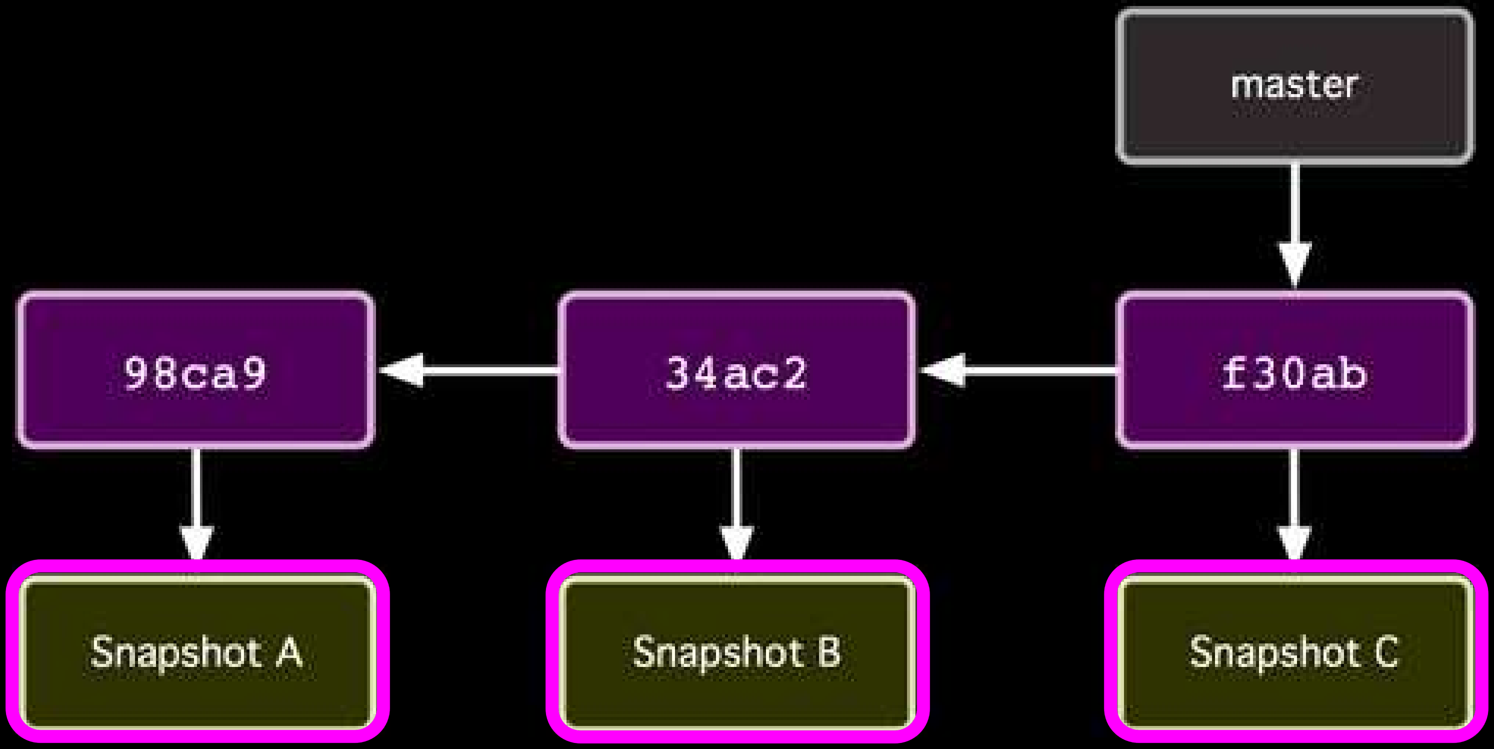
Repo Organization

Commits (from oldest to newest; hashes as commit IDs)



http://git-scm.com/book/

Repo Organization

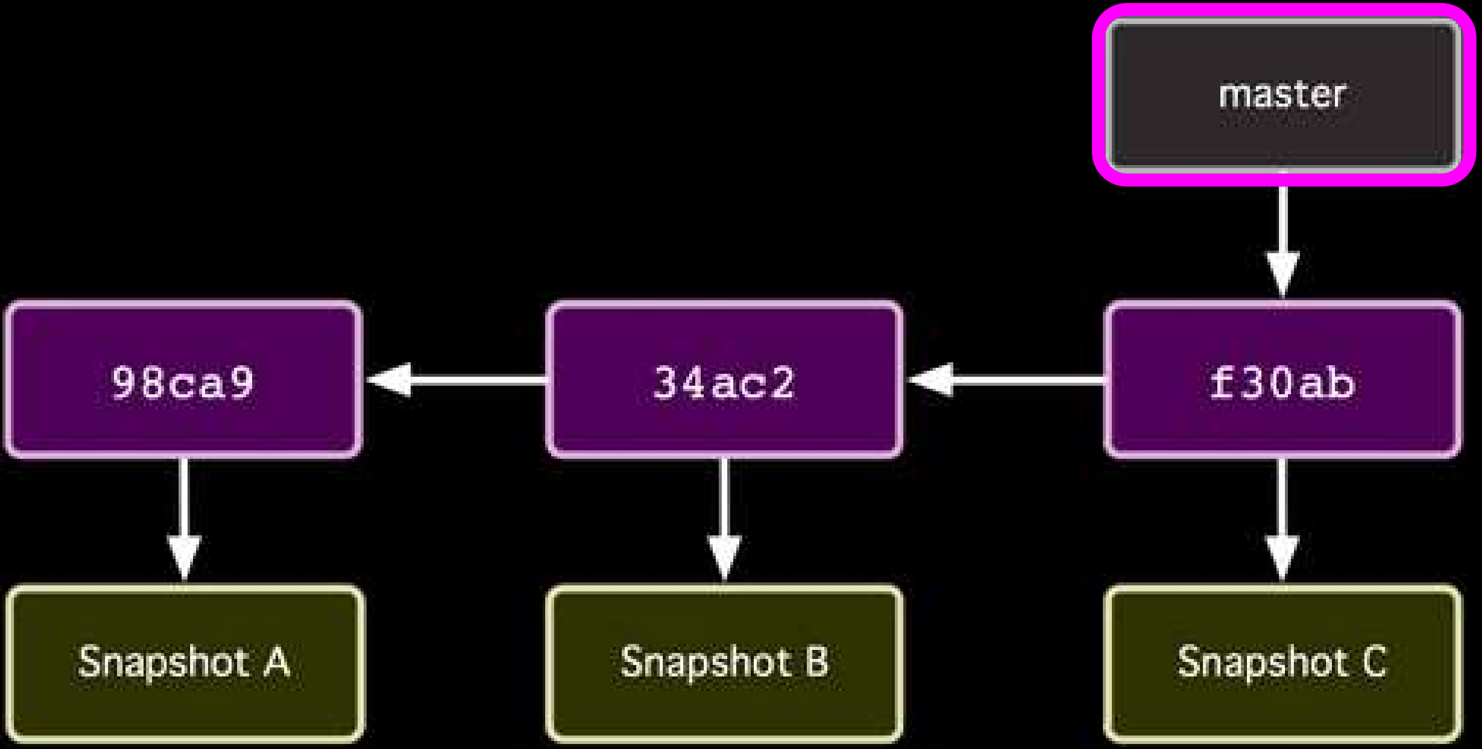


Snapshot of all files at each commit

http://git-scm.com/book/

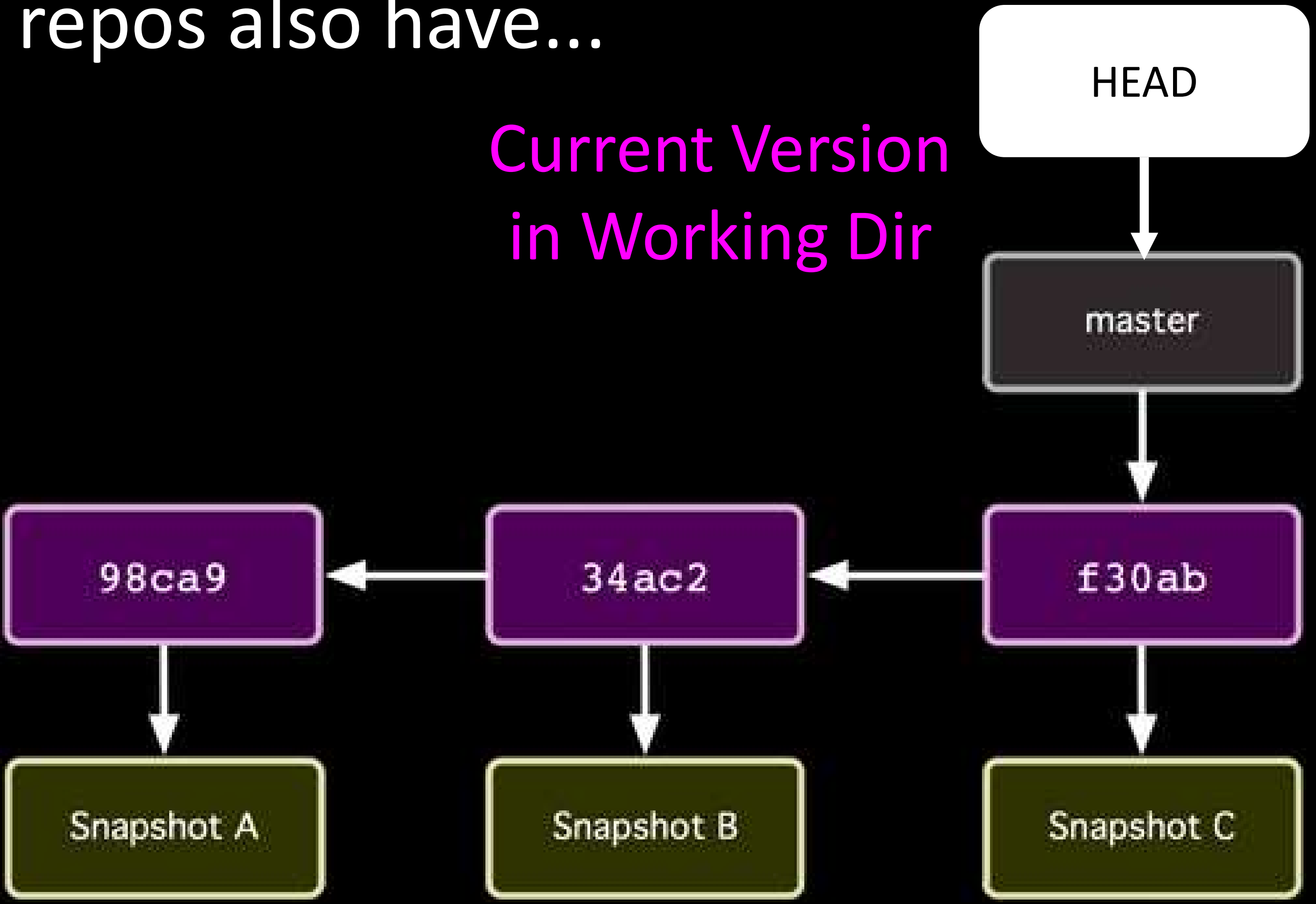
Repo Organization

Branch (last commit)



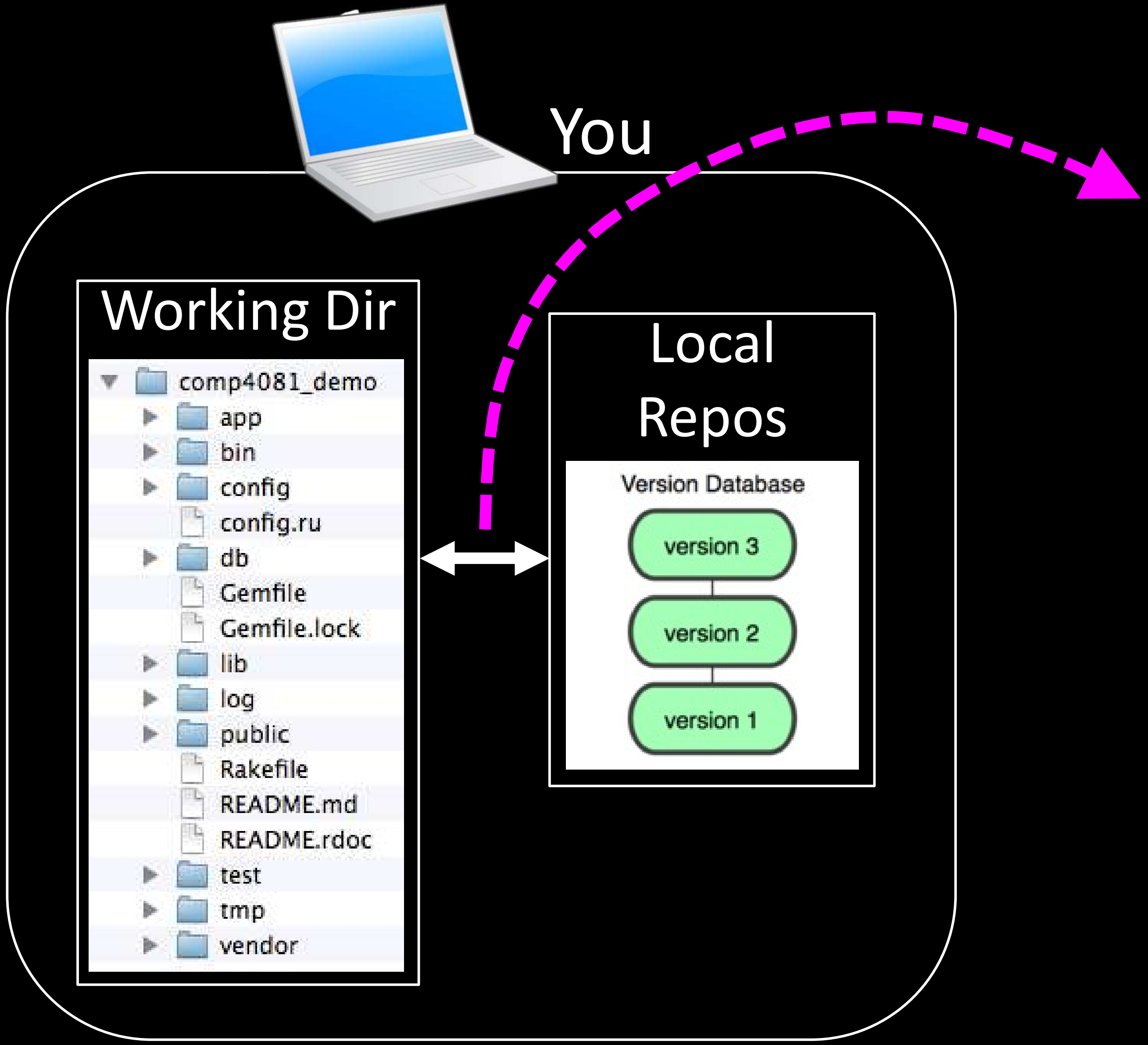
http://git-scm.com/book/

Local repos also have...



http://git-scm.com/book/

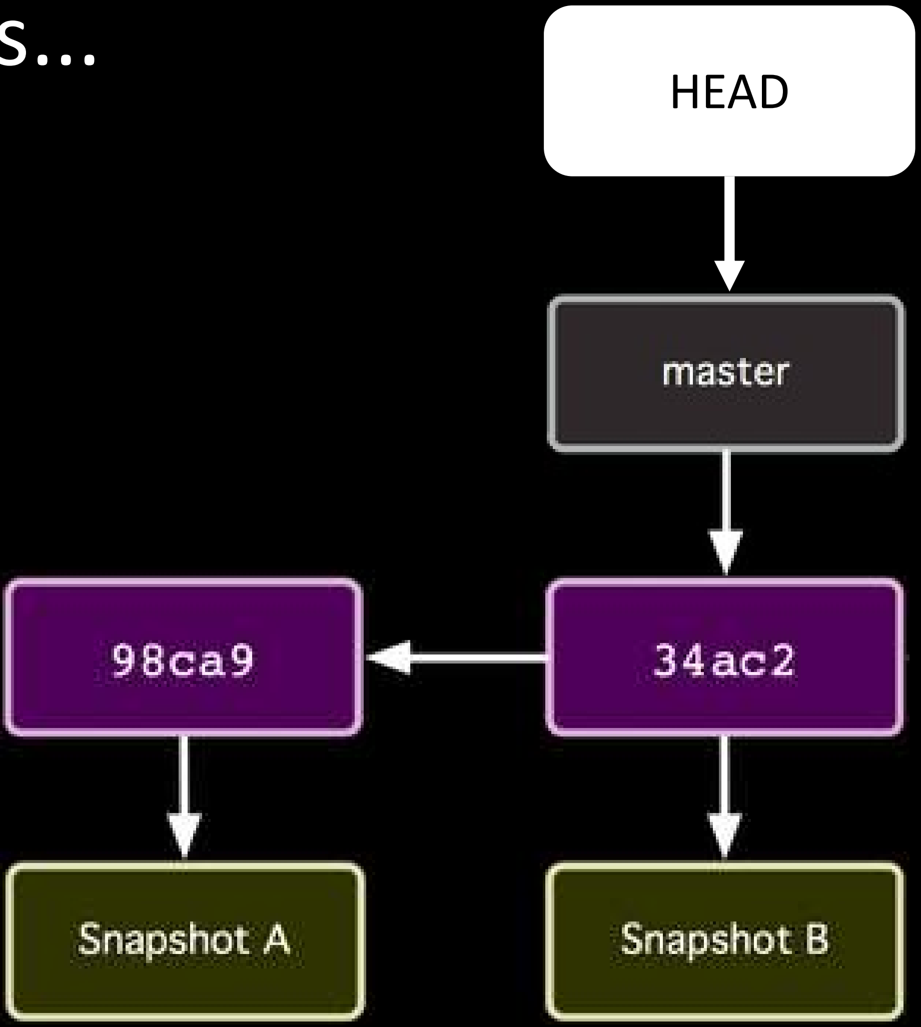
Local Repo Operations



- init
- add/commit
- log
- switch/checkout
- branch
- merge
- …

How commit works...

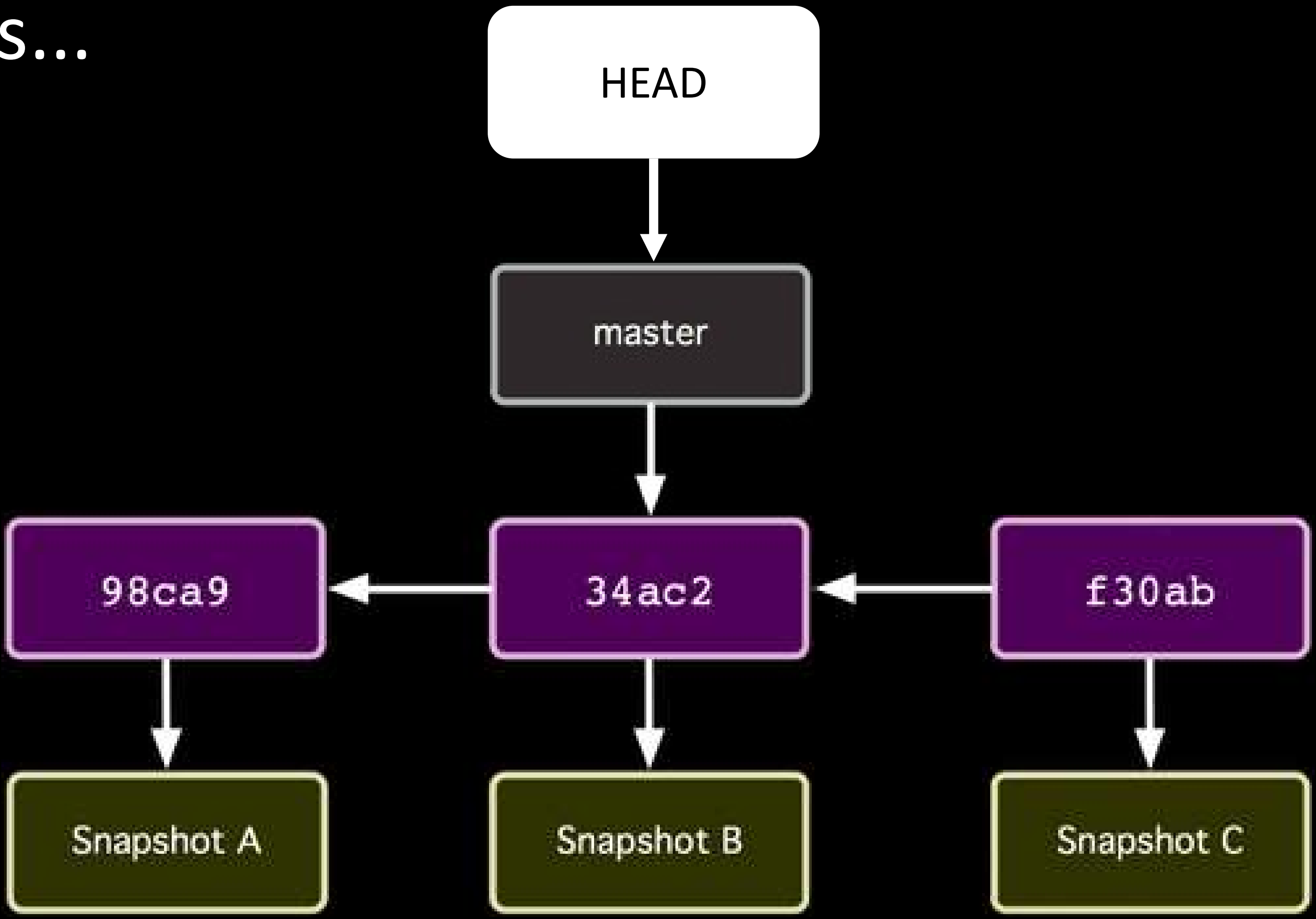
Before



http://git-scm.com/book/

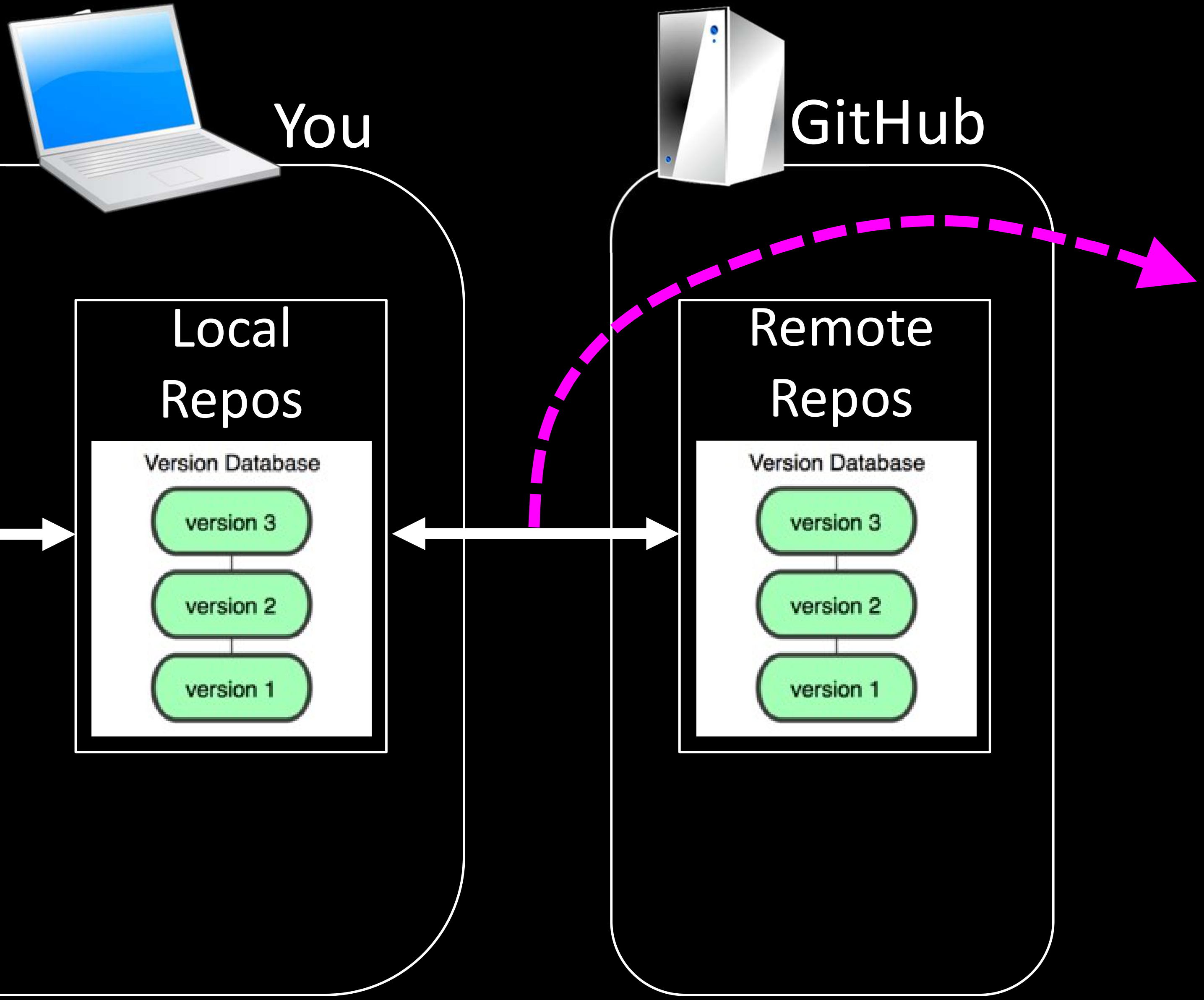
How commit works...

After



http://git-scm.com/book/

Remote Repo Operations



- clone
- push
- pull
- fetch
- remote
- …

Learn the Core Local/Remote Operations

- Demo Videos:
  - https://scott-fleming.github.io/web-dev-rails-git-tutorial/demo-04-git-local.html
  - https://scott-fleming.github.io/web-dev-rails-git-tutorial/demo-05-git-branches.html
  - https://scott-fleming.github.io/web-dev-rails-git-tutorial/demo-06-git-remote.html

Summary

- Version Control Systems
- Git and GitHub
- Repo Structure
- Local/Remote Repo Operations
- Commit Semantics

http://flic.kr/p/aCLor3