Version Control with Git

- Before we start
  - Sign up at github.com

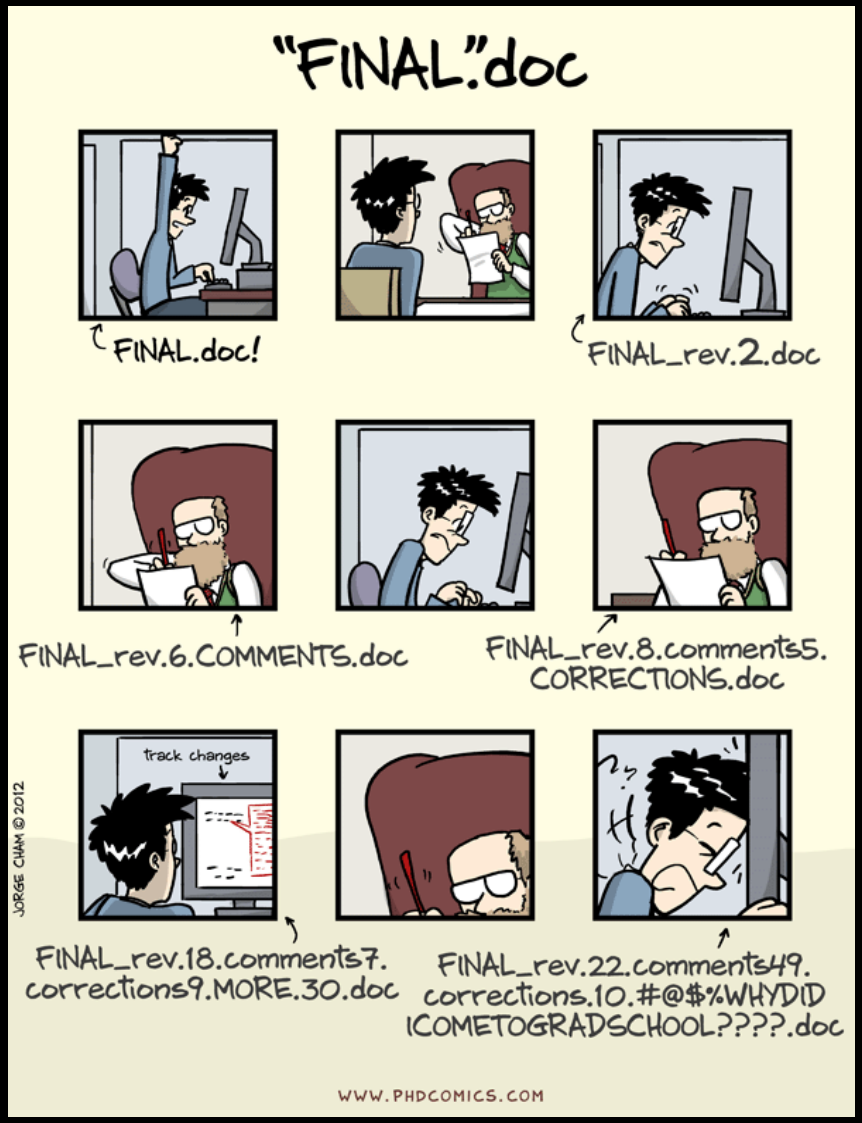
What is Version Control?

(AKA revision control, source control)

- Tracks changes to files
- Any file can be tracked
- Text (.txt, .csv, .py, .c, .r etc.) works best
    - These allow smart *diff* / *merge* etc.

Why Use Version Control? #1

- A more efficient backup
- Reproducibility



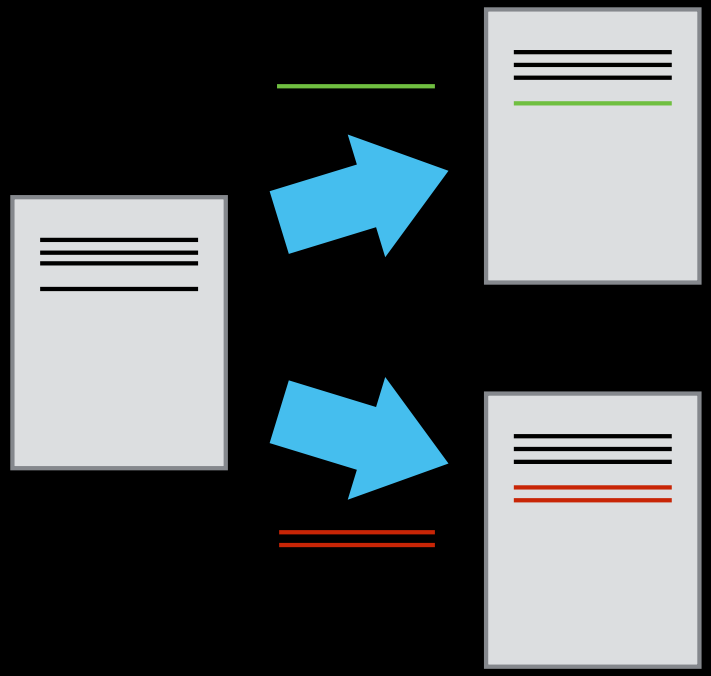
Why Use Version Control? #2

- Teamwork

Version Control Tracks Changes

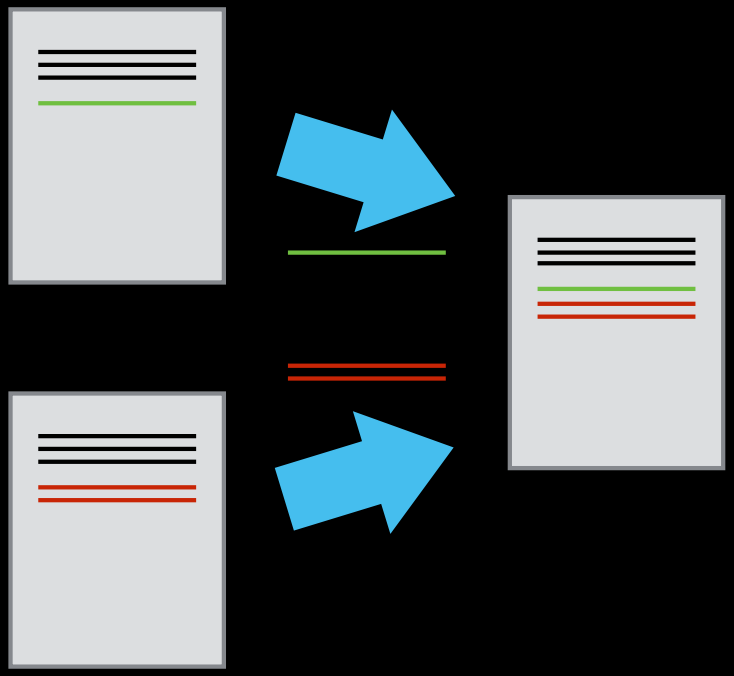
- Changes are tracked sequentially

Version Control Tracks Changes



- Different versions can be saved

Version Control Tracks Changes



- Multiple versions can be merged

Version Control Alternatives

- Subversion (svn) - Centralised
- Mercurial (hg) - Distributed
- Git (git) – Distributed

- N.B. GitHub != git

Local Configuration

- git config

Getting Demo Files

- git clone https://github.com/Southampton-RSG/2019-11-19-southampton-swc

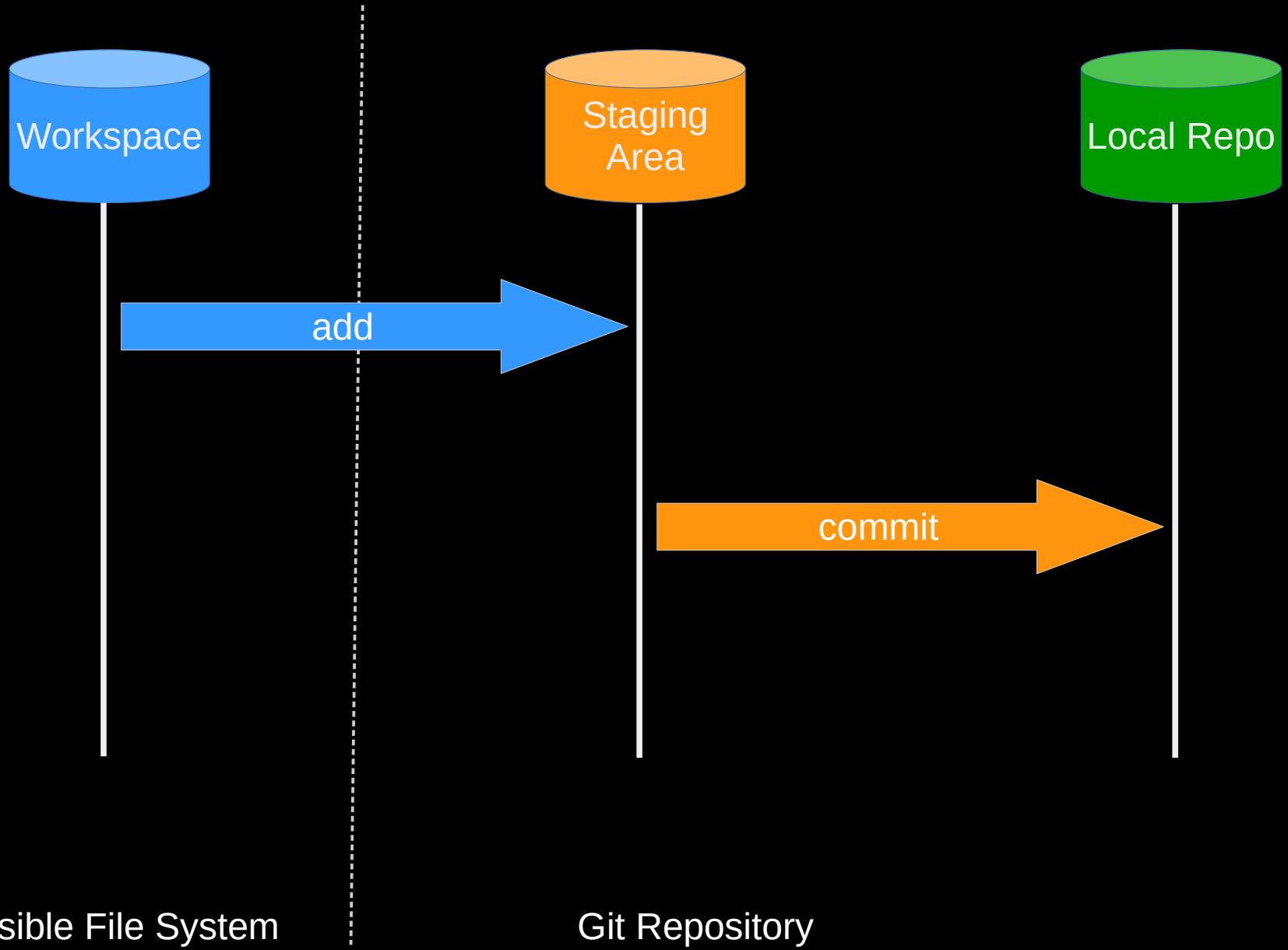
Creating a Repository

- git init
- git status

Tracking Changes to Files

- git add
- git commit

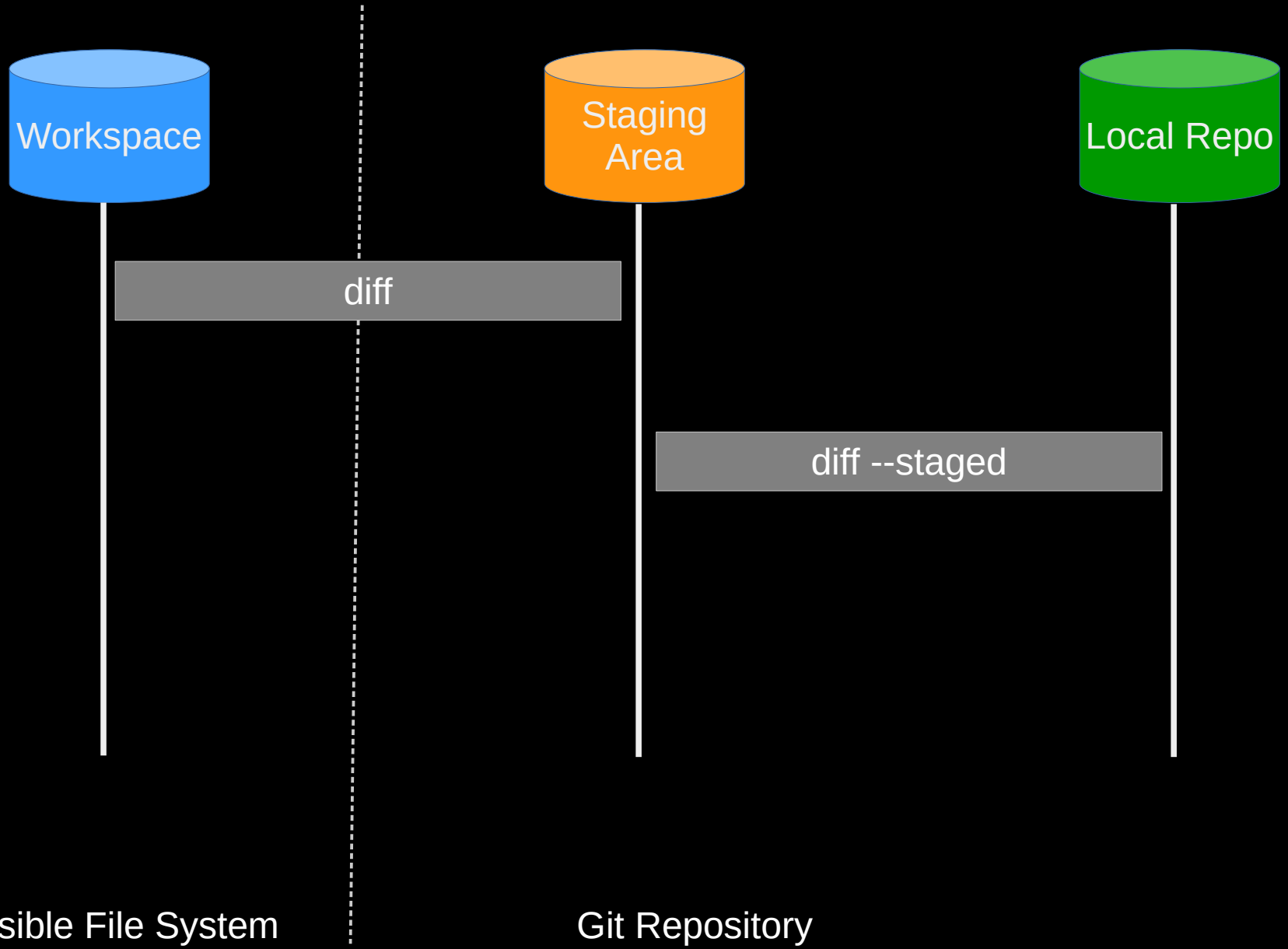
Git – add and commit



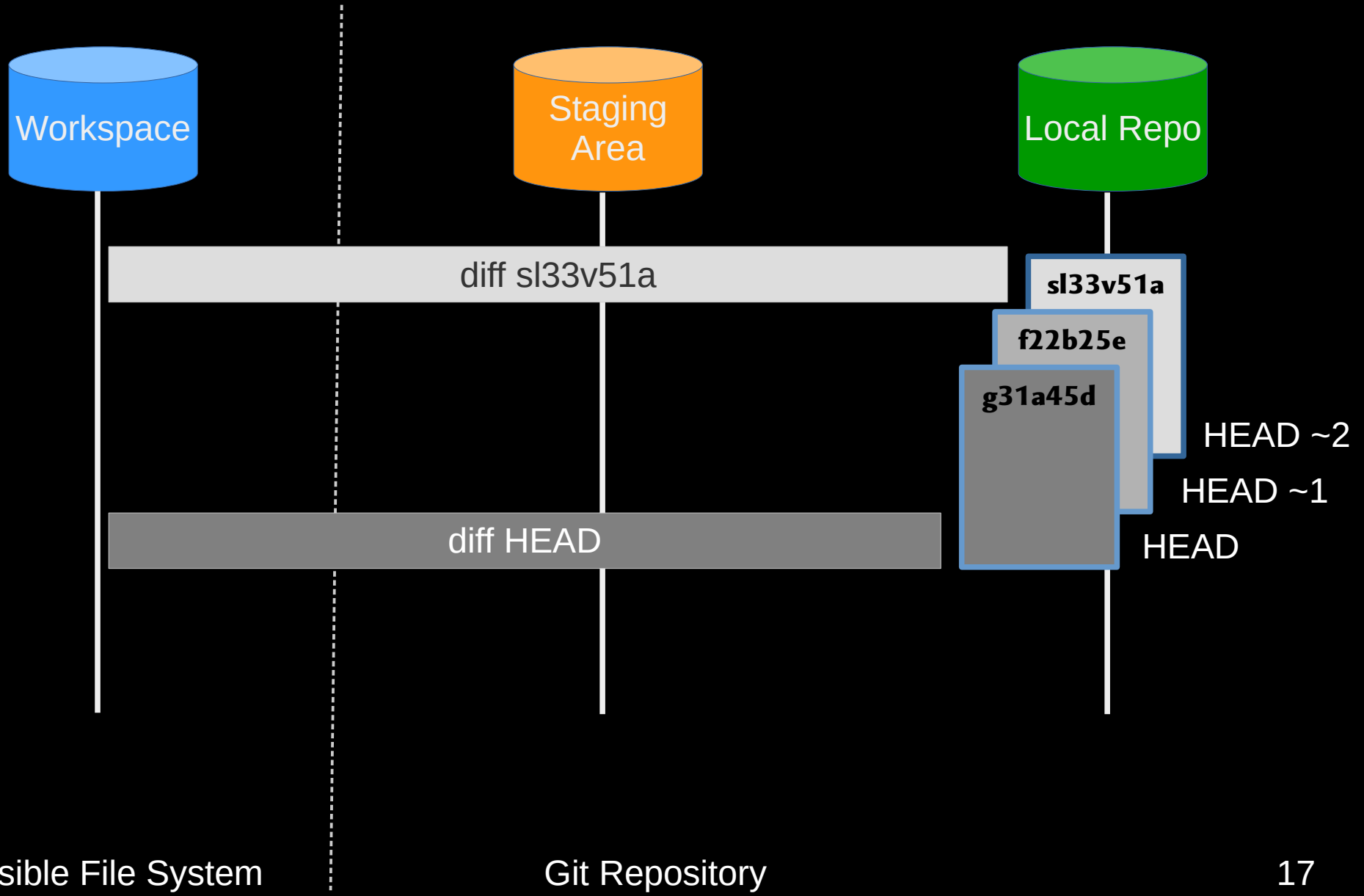
Exploring History #1

- git log
- git diff

Git – diff #1


Workspace
Staging Area
Local Repo
diff
diff --staged
Visible File System
Git Repository

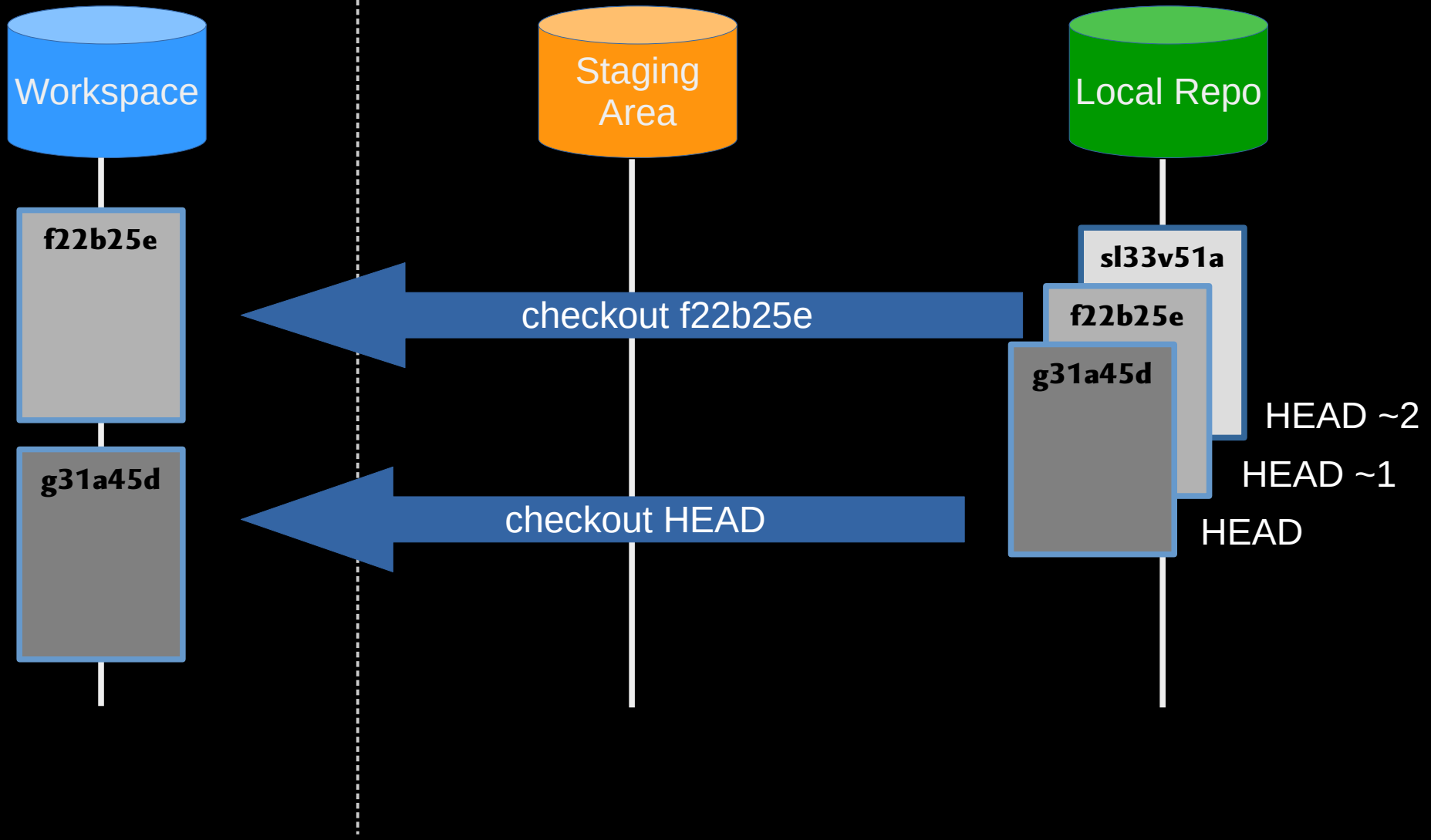
Git – diff #2


Workspace
Staging Area
Local Repo
diff sl33v51a
diff HEAD
sl33v51a
f22b25e
g31a45d
HEAD ~2
HEAD ~1
HEAD
Visible File System
Git Repository

Restoring Files

- git checkout

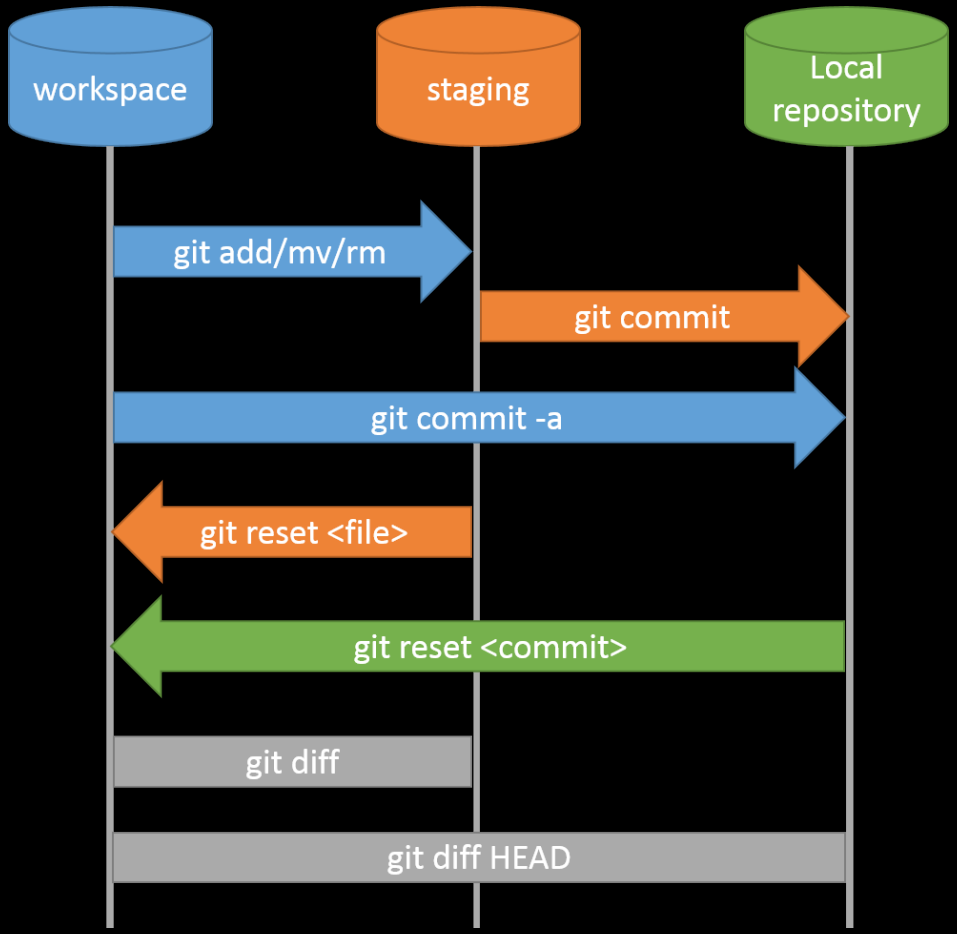
Git - restoration



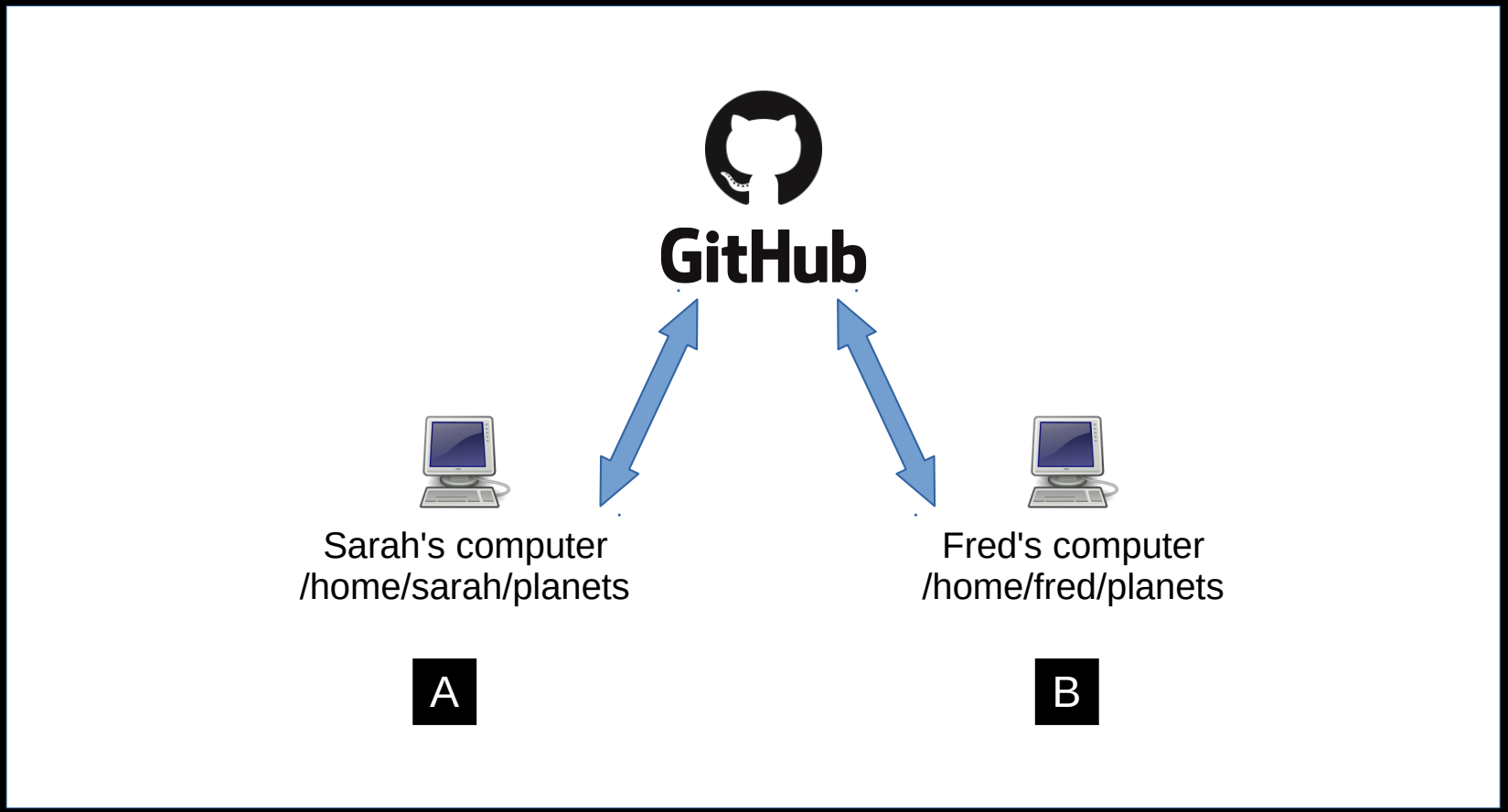
Visible File System

Git Repository

Git Workflow – Local Repo.



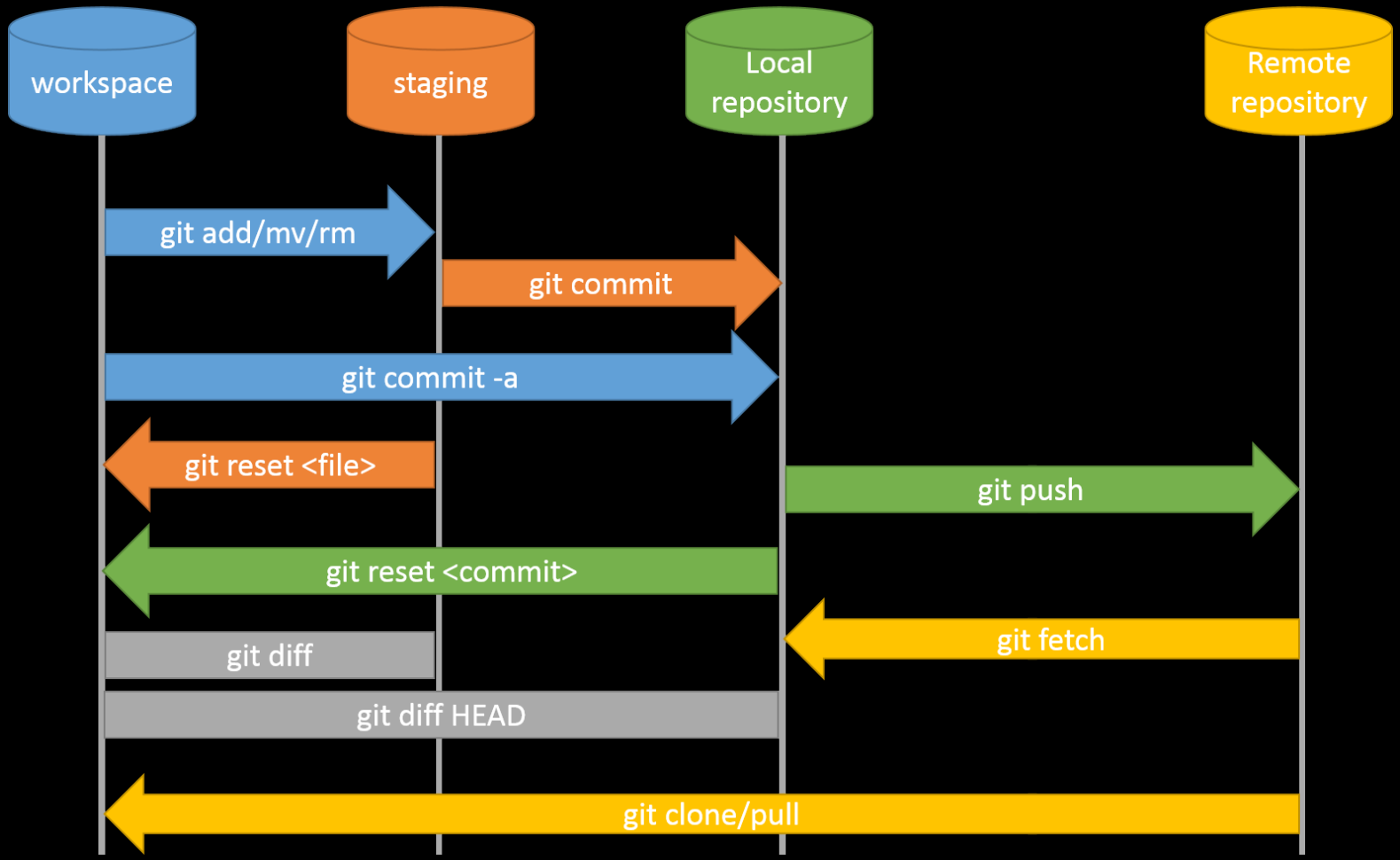
Collaboration



Collaboration: Remote Repositories

- Sign in https://github.com/
- Create repository
- git remote add
- git push

Git Workflow – Remote Repo.



Collaboration: Feature Branch Exercise

- Check out ‘dev’
- Create a new branch called ‘docs’
- Create and add README.md
- Push to GitHub and merge back to ‘dev’
- Pull the changes back to your computer

What next?

- Ignore files / Merging
- https://software-carpentry.org