Git Basics

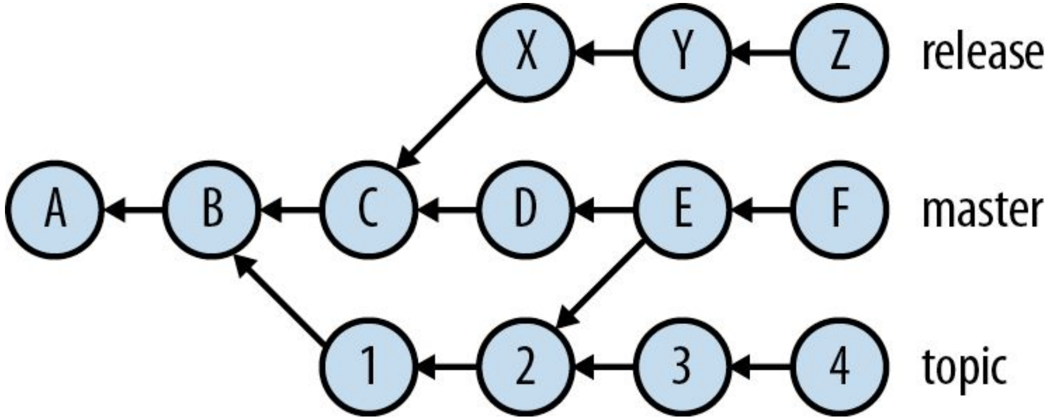
Benefits

- In-built Code backup
- Code snapshot
- Automation
- Code Hygiene

Available for free by the magic of open-source.

Git Intuition

- Directed Acyclic Graph (DAG) of commits
- DAGs are built using linked lists
- Each commit points to its parent



Installation - MacOS

- Install Homebrew - brew.sh (follow the link)
- `brew install git`
- `git --version` (verify installation)

Repository

- What is Repository?
- Local Repo
- Remote Repo
- Create Repo

Create Repository - Remote to Local

1. Create remote repo on Bitbucket/Github (through browser)
2. Clone on your machine - done on terminal (local repo)

```
git clone <remote-repository-url>
```

3. No code = No Repo DAG
4. Start coding

Create Repository - Local to Remote

1. Create on your machine (local repo)

   ```
   git init
   ```

2. Start coding
3. `git add` and `git commit`
4. Create remote repo on Bitbucket/Github (through browser)
5. Update remote url in the local repo (remote origin)

   ```
   git remote add origin <remote-repository-url>
   ```

6. Push from local to remote

   ```
   git push -u origin main
   ```

Git Add

- Working directory
- Staging area or index
- Command to add files

Preparing for Code Snapshot

- Write code in your current directory
- Working directory - directory with `.git` [need]
- Staging area or index [need]
- Add files to the staging area

```
git status

git add .

git add <file_name>

git status
```

- No snapshot = No Repo DAG

Git Commit

- What is a commit?
- Command to create commit
- Commit message

Saving Code Snapshot

- Commit = current code snapshot
- Snapshot = current state of ALL files in the staging area
- Revisiting repository definition (hint: `.git`)
- Create commit:

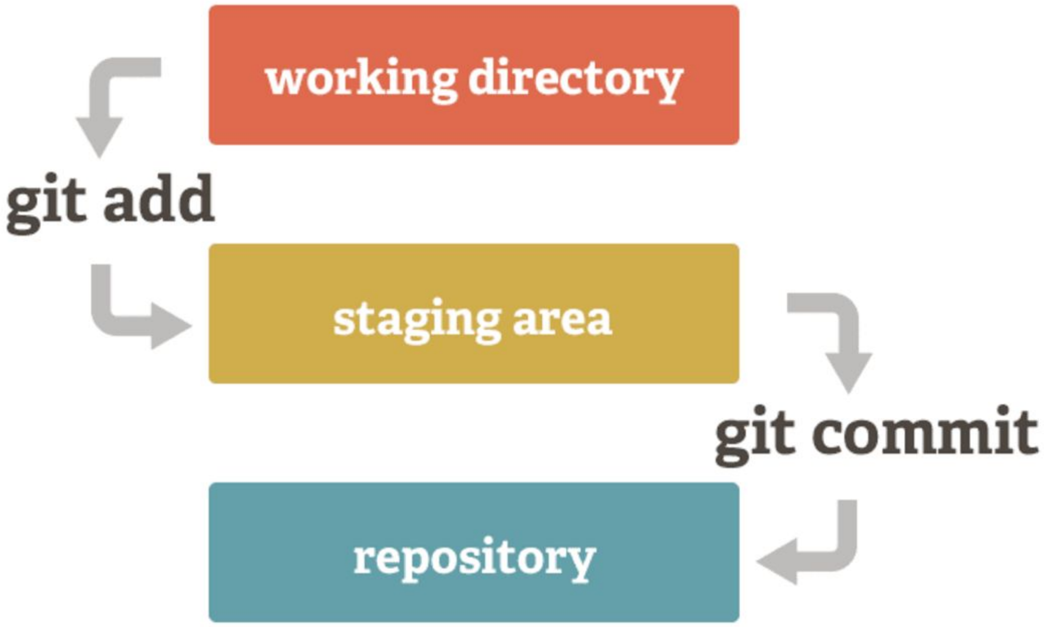
```
    git commit -m “Add: my new shiny class”
```

- Commit message: Label for each snapshot
- Seeing your commit:

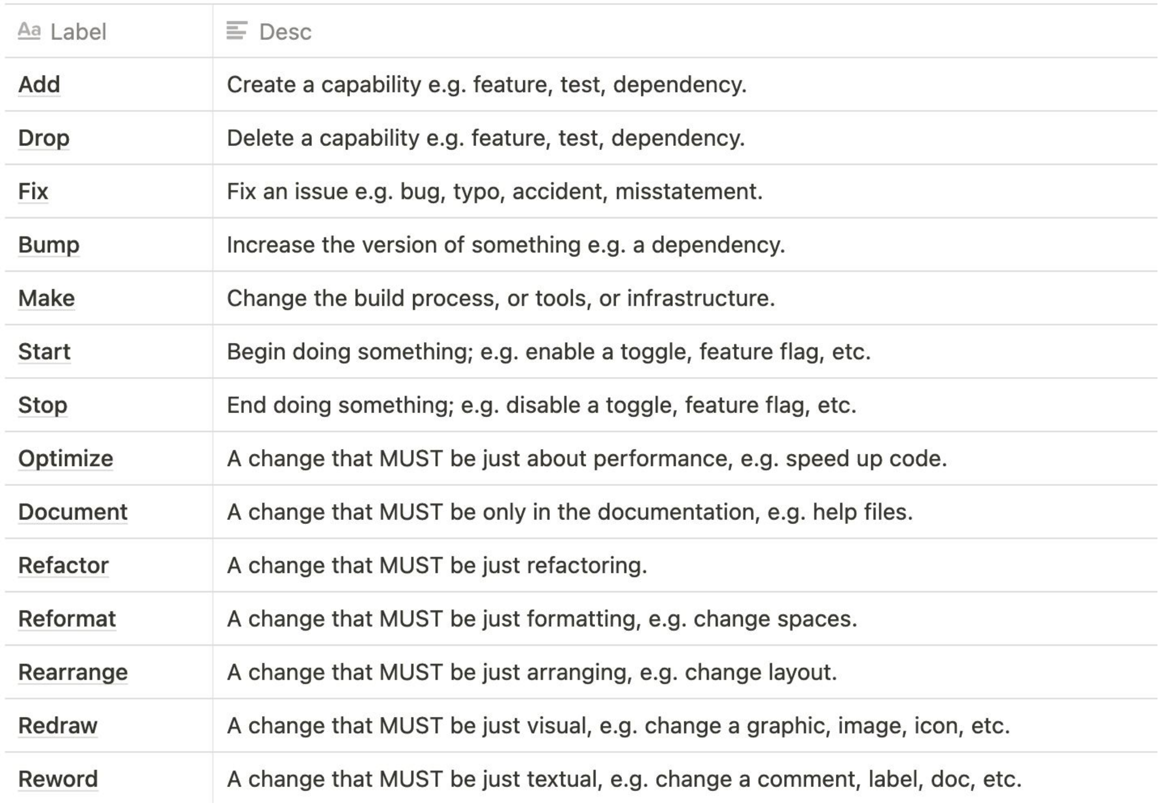
```
    git show <commit_hash>
```

- 1 Snapshot = 1 node DAG

Three Worlds of Git



Commit Message Guide



Label	Desc
Add	Create a capability e.g. feature, test, dependency.
Drop	Delete a capability e.g. feature, test, dependency.
Fix	Fix an issue e.g. bug, typo, accident, misstatement.
Bump	Increase the version of something e.g. a dependency.
Make	Change the build process, or tools, or infrastructure.
Start	Begin doing something; e.g. enable a toggle, feature flag, etc.
Stop	End doing something; e.g. disable a toggle, feature flag, etc.
Optimize	A change that MUST be just about performance, e.g. speed up code.
Document	A change that MUST be only in the documentation, e.g. help files.
Refactor	A change that MUST be just refactoring.
Reformat	A change that MUST be just formatting, e.g. change spaces.
Rearrange	A change that MUST be just arranging, e.g. change layout.
Redraw	A change that MUST be just visual, e.g. change a graphic, image, icon, etc.
Reword	A change that MUST be just textual, e.g. change a comment, label, doc, etc.

Git Push

- Upload history to cloud
- Command to push to cloud

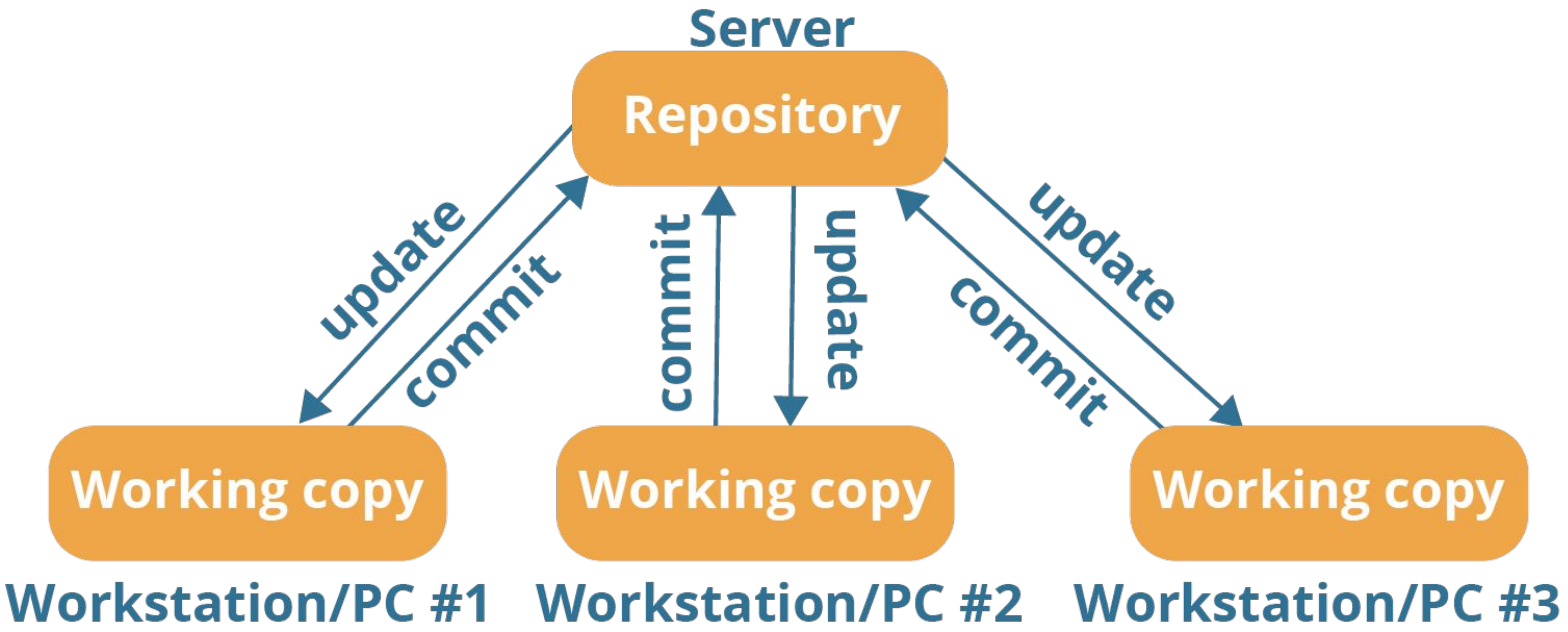
Remote History

- Maintain backup
- Collaboration
- Pushing to remote

```
git push origin main
```

- `origin` = remote repository
- `main` = current branch (local repository)

Centralized version control system

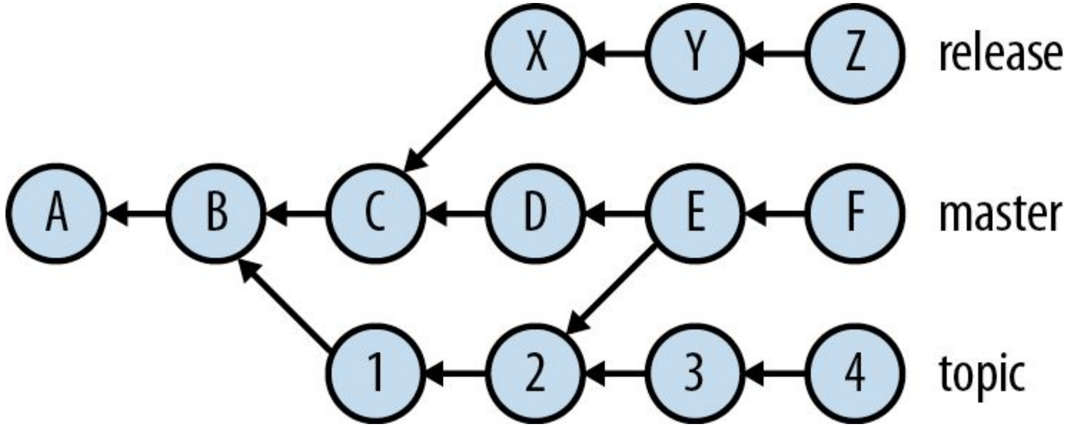


Git Branch

- What is a branch?
- Default branch

Branch

- Special name for a commit
- Switching to a branch = loading a particular code snapshot
- Check the branch history
    - `git log --oneline --decorate --graph`



Next Steps

- Create your repos on Bitbucket or Github
- Connect it with DataBricks
- Add, Commit, and Push to the repo from DBR
- Clone to local
- Add, Commit, and Push to the repo from local

Be on the lookout for the next Git session