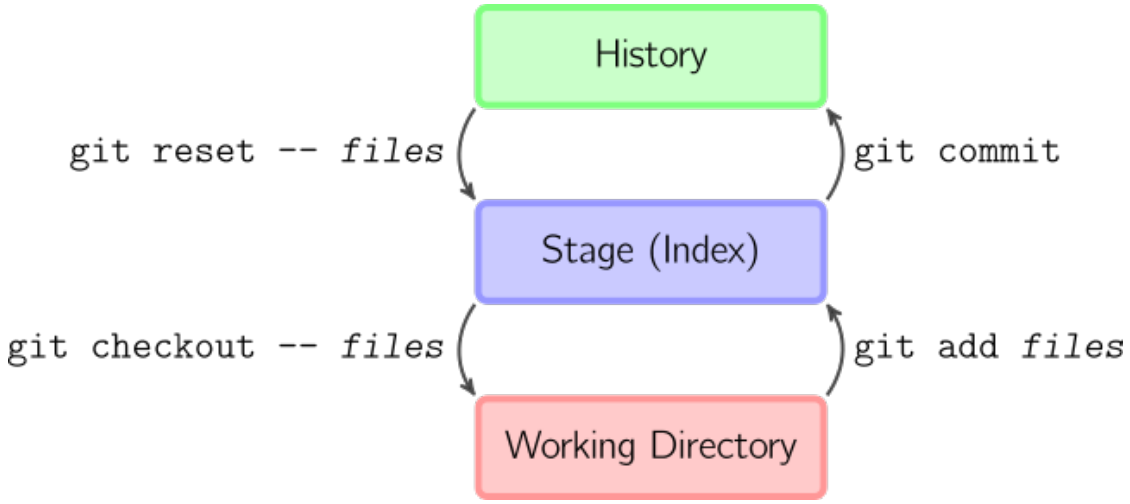
Initialize a Repository

Hamid Semiyari

Basic Git

- A **repository** (or repo, for short) is a collection of files (in a folder and its subfolders) that are together under version control. In data analysis, each repository is typically one project (like a data analysis, a homework, or a collection of code that performs a similar task).
- The way git works (graphic from Mark Lodato):



- **Working Directory**: To git, this means the current versions of the files. Changes to files that you haven't recorded only exist in the working directory and are not yet saved in the history.
- **Stage**: Files that are scheduled to be committed to the history, but not yet committed. Only files in the stage will be committed to the history.
- **History**: The timeline of snapshots of files. You commit a file to the history and then, even if you modify it later, you can always go back to that same file version.

- We'll focus on the right-hand-side of this diagram where your workflow is typically:

    1. Modify files in your working directory until you want a snapshot.
    2. Add these modified files to the staging area.
    3. Commit staged files to history, where they will be kept forever.

- The left-hand side of the diagram is used when you want to undo mistakes.
- All git commands begin with `git` followed immediately by an argument for the type of command you want to execute.
- For the right-hand-side of the diagram, the following are the useful git commands:

    - `git init`: Initialize a git repository. *Only do this once per project.*
    - `git status`: Show which files are staged in your working directory, and which are modified but not staged.
    - `git add`: Add modified files from your working directory to the stage.
    - `git diff`: Look at how files in the working directory have been modified.
    - `git diff --staged`: Look at how files in the stage have been modified.
    - `git commit -m "[descriptive message]"`: commit your staged content as a new commit snapshot.

Initialize a repository

- Git needs to be told that a folder is a repo. Otherwise, it won't keep files under version control.
- In this class, you won't need to tell git this (I'll tell git this), but in the real world you will. So we'll go over how to do this on GitHub and on the terminal.

On the terminal

- Don't initialize on your local for this lecture. These are just the steps you would do if you needed to initialize on your local.
- Use `cd` to enter the folder that you would like to keep under version control.
- The use `git init`

```
git init
```

- This will tell git that the folder is a single repo.
- Your files are not yet tracked. You'll need to do the steps below to tell git which files to track. But at least git now knows that this is a repo where tracking is possible.

On GitHub

- Git is a version control system, GitHub is a website that hosts git repositories. (so on your resume, say that you know git, not GitHub).
- You can create a git repo on GitHub (GitHub's server is called the "remote"), then download ("clone") the repo onto your computer (your computer is called the "local").
- On your GitHub homepage, click on "New"
- Fill out the form. The options are pretty self-explanatory, and GitHub does a good job of providing descriptions. For this lecture, make sure
  - Repository name is "test".
  - The repo is set to be "Private"
  - You check "Add a README File"
- Click on "Create Repository.