Intro to Git

Jon Ambler

What is Git?

- A version control system (VCS)
  - Git, Subversion, Perforce
- What is a VCS?
  - Keeps multiple version of the same file
  - Allows you to undo changes

What is Git?

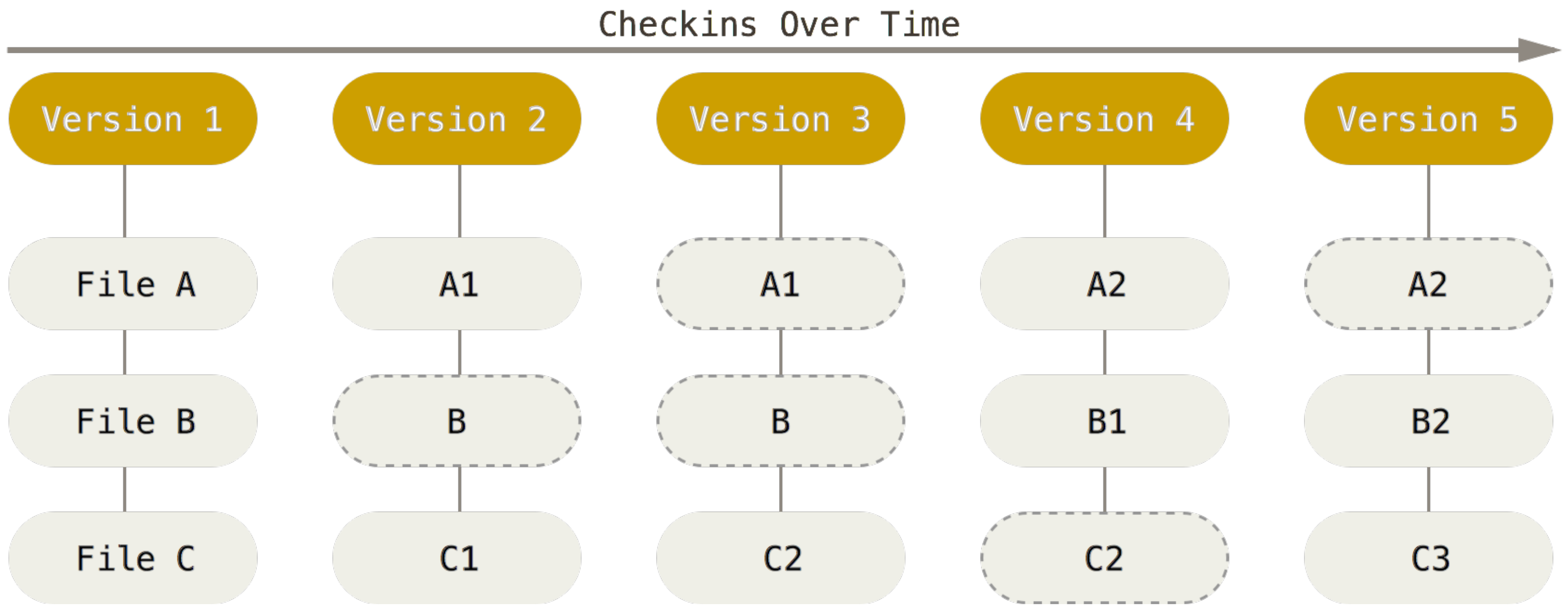
- Used for:
  - Working collaboratively on the same project
  - Keeping work safe
  - Keeping multiple versions of the same file

What is GitHub?

- Also BitBucket
- Now owned by Microsoft

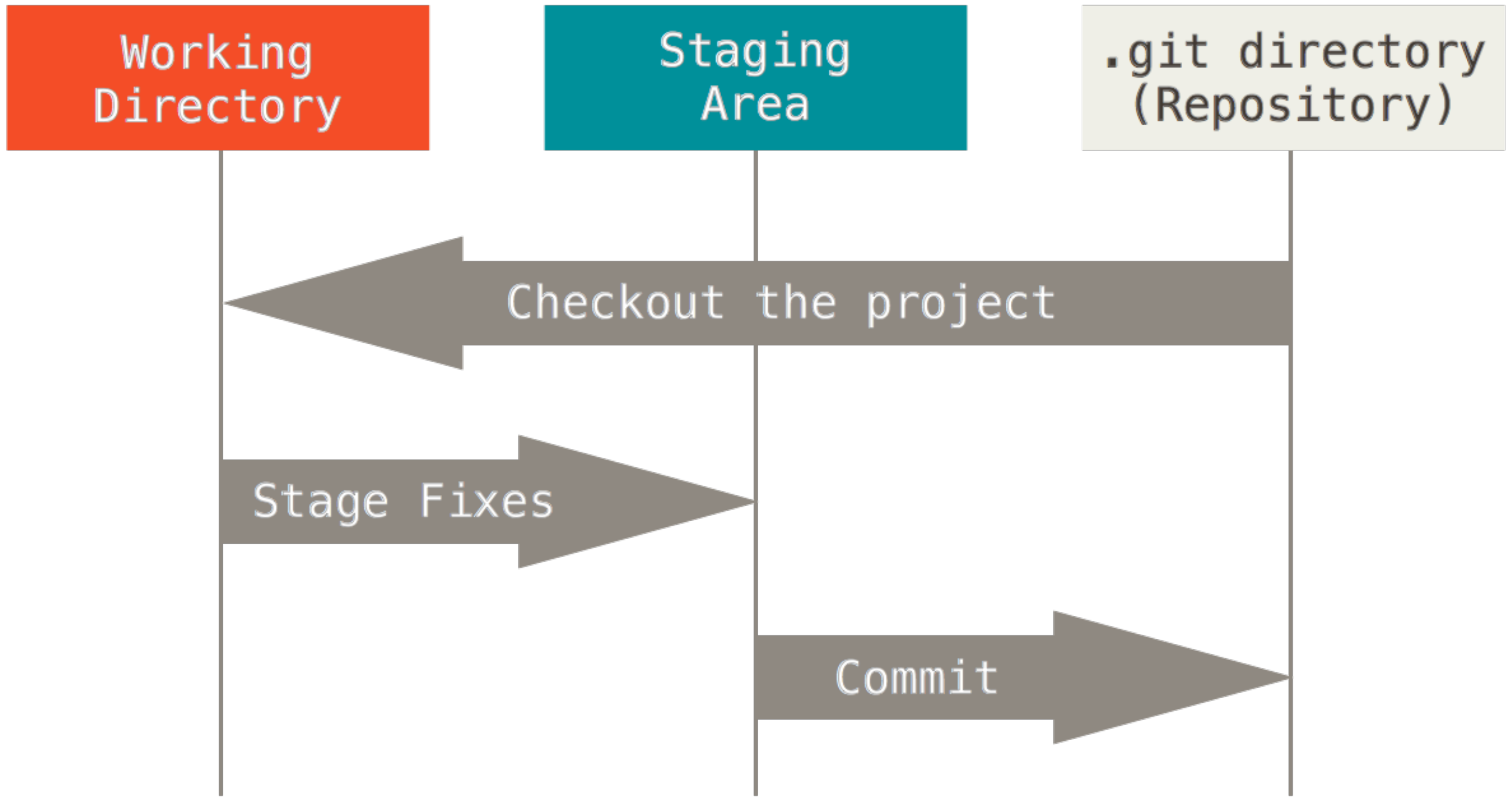
What is Git?

- Git stores streams of snapshots
- Other VCS systems store changes



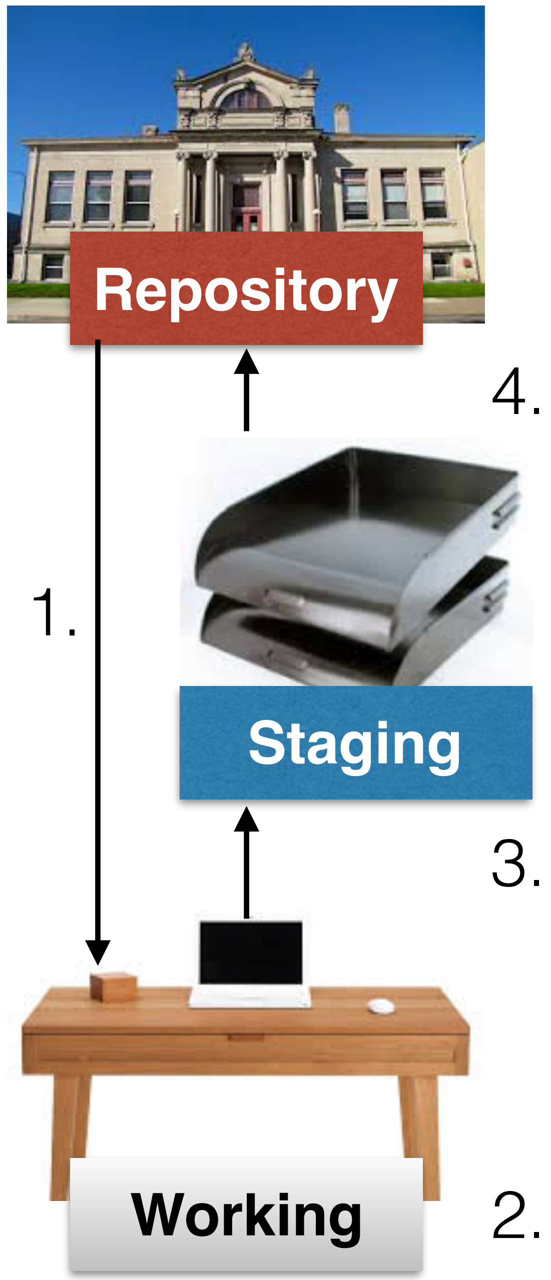
Three states in Git

- A file can be committed, modified, or staged



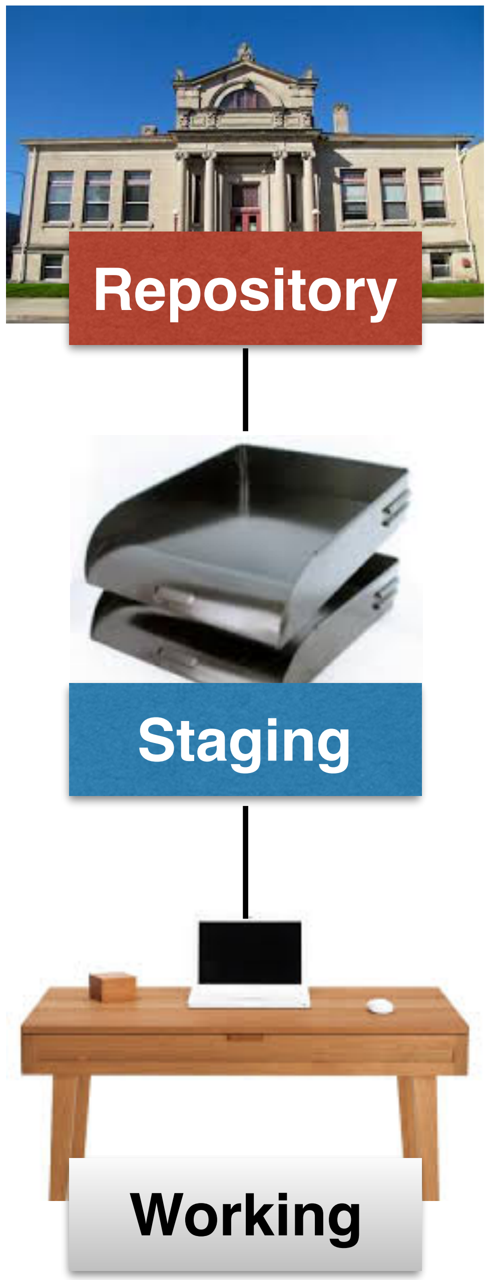
Git workflow

1. Pull latest version from the repository
2. Modify the files in your working directory
3. Stage the files you have modified
4. Commit the files in your staging area to your Git repository



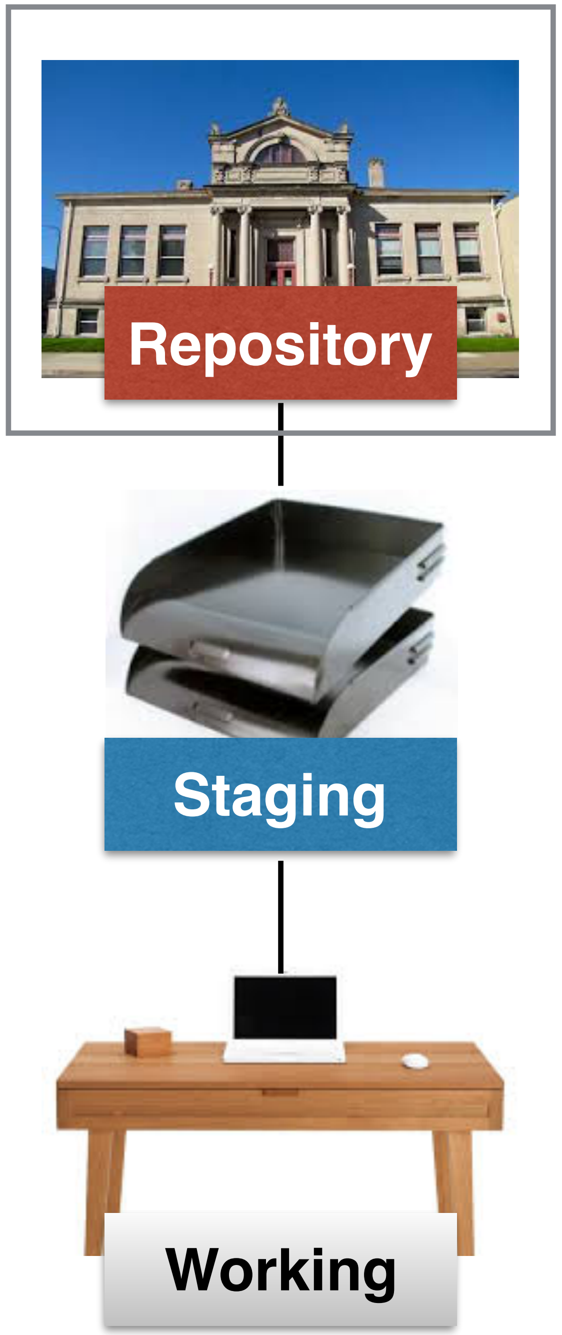
Git usage

- Can be used from:
  - The command line
  - A browser
  - A GUI
  - Built into interactive development environments (IDEs)



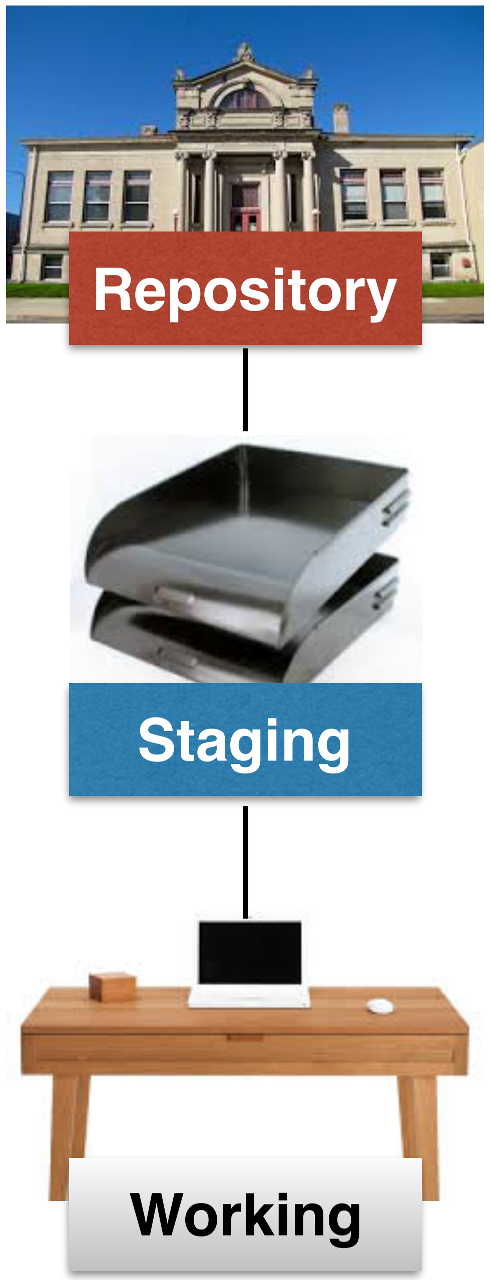
Repositories

- Local
  - More control
  - For sensitive data
- Remote
  - Github, Bitbucket
  - Safer backup-wise
  - Files are not private (Free version)



Startup: First time

- `$ git config`
  - Configuration stored at 3 levels
    - System (`--system`)
    - User (`--global`)
    - Directory



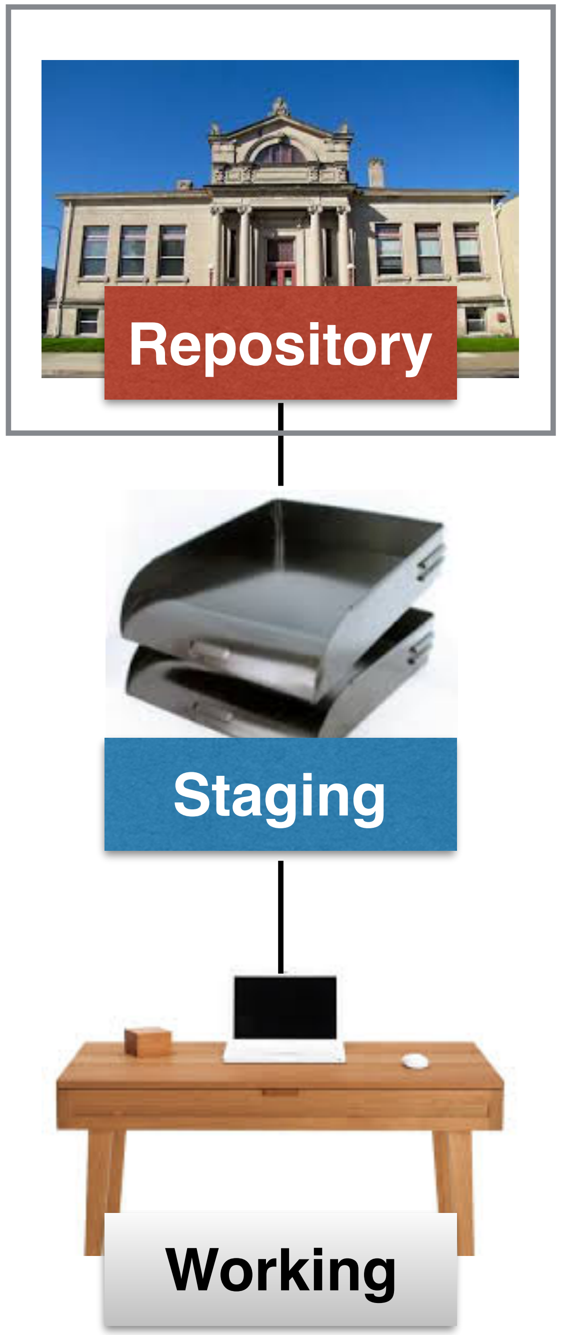
Startup: First time

- Setting the user identity
    - `$ git config --global user.name "John Doe”`
    - `$ git config --global user.email johndoe@example.com`
    - These must be the same as your Github details

Repository
Staging
Working

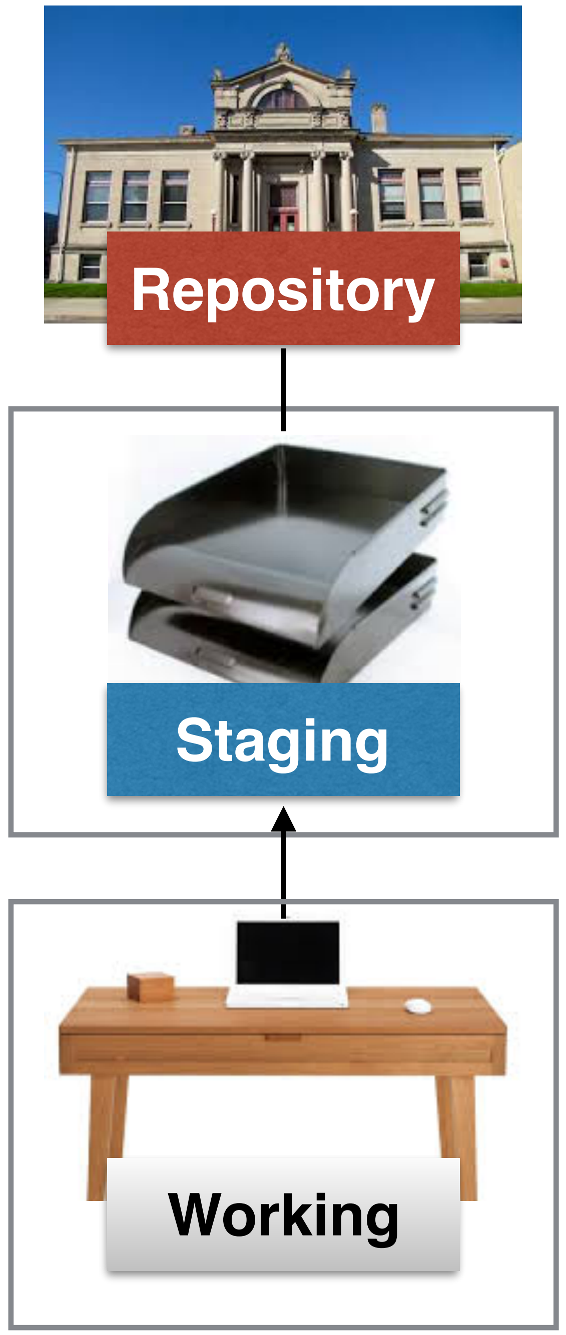
Creating a repository

- Creating a local repo:
  - `$ git init`
- Creating a remote repo:
  - https://github.com



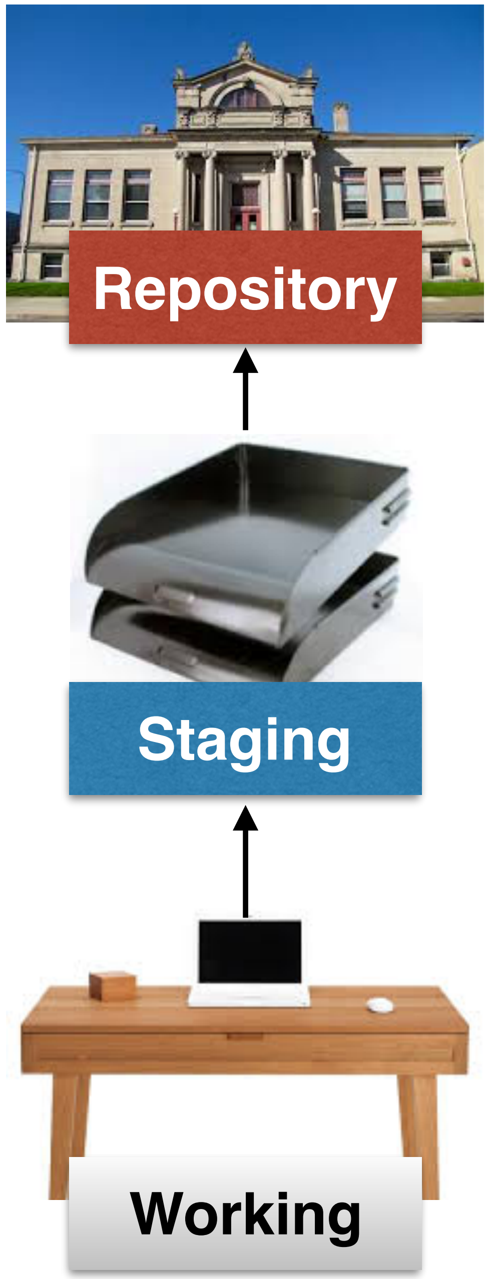
Adding files for VC

- Add files to be tracked:
    - `$ git add <file>`
- Checking what is the state of the files in the dir
    - `$ git status`



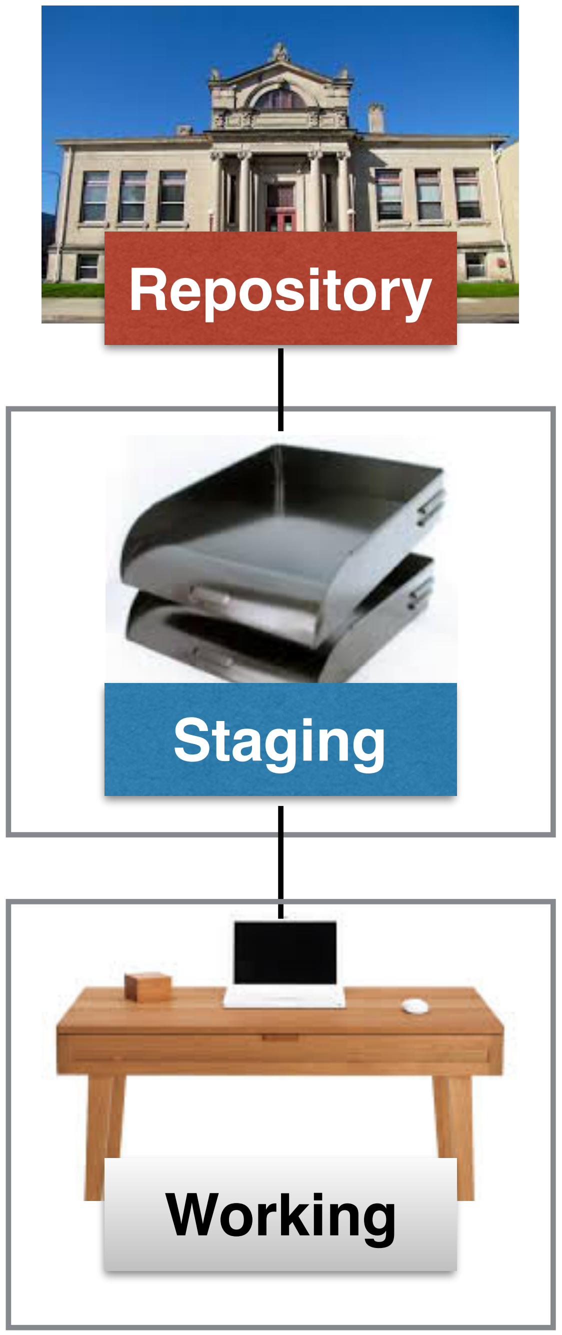
Git usage

- To move a file to the staging area from the working area
  - `$ git add <file>`
- To move a file from the staging area to the local repository
  - `$ git commit`



Git usage

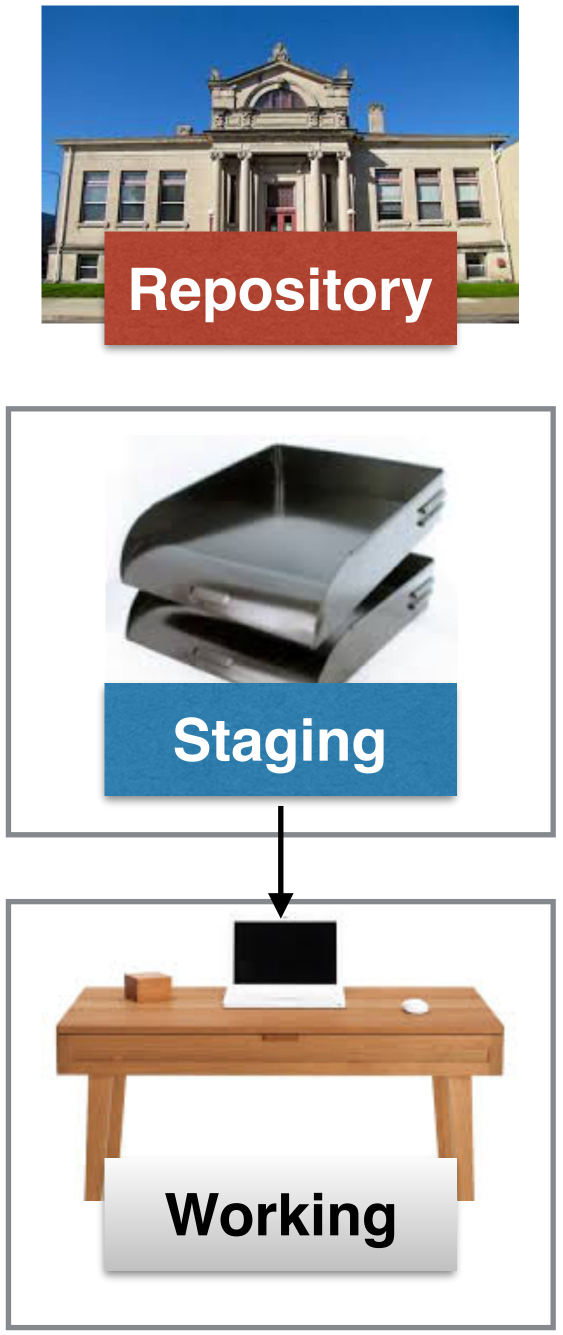
- If you start working on a staged file:
  - You will be working on a different file to the one that is staged
  - If you run git commit, the staged file, not the one you are working on will be committed to the repo



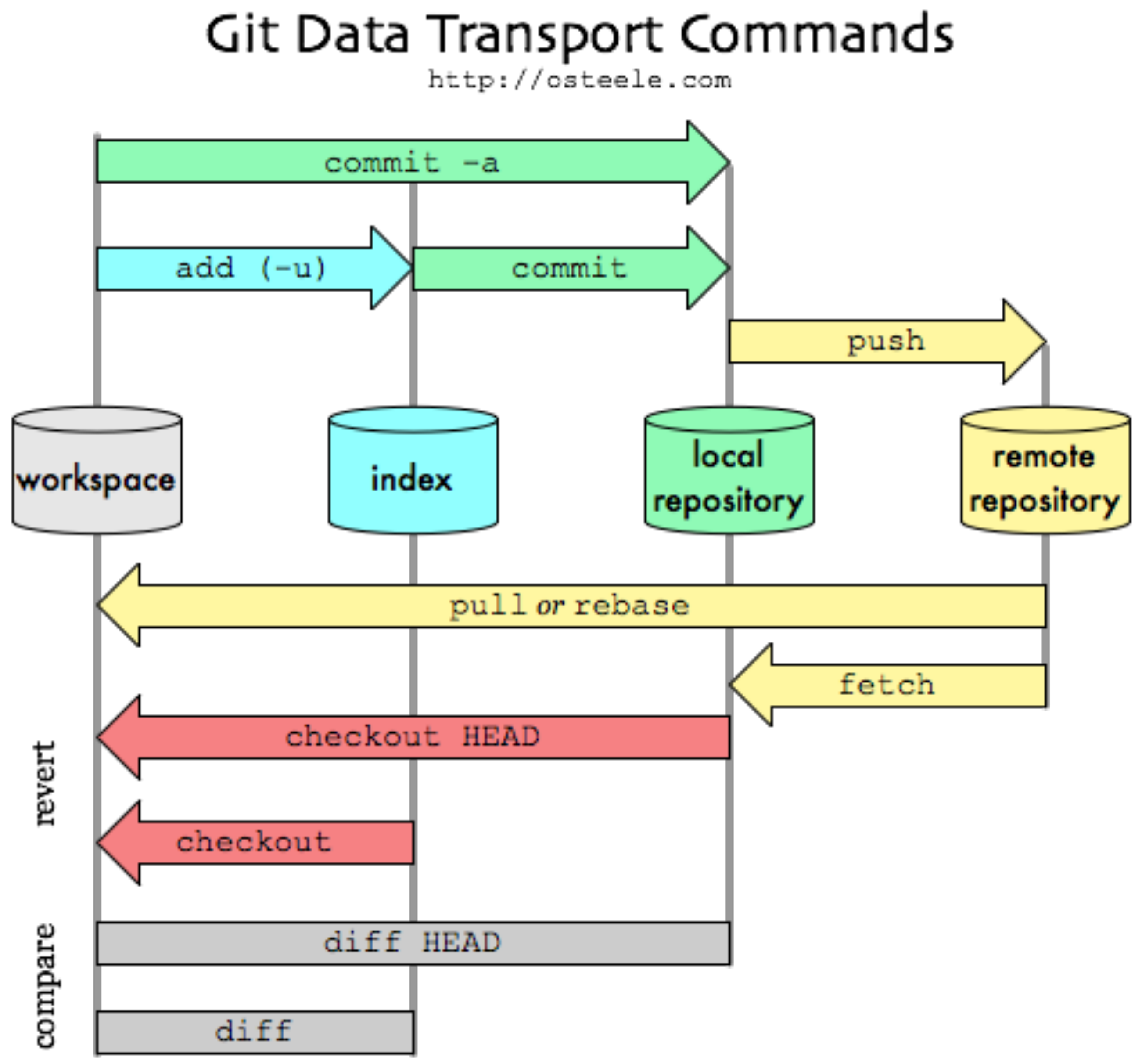
Git usage

- To undo the changes you have made and revert to a previously committed version

  - `$ git checkout`

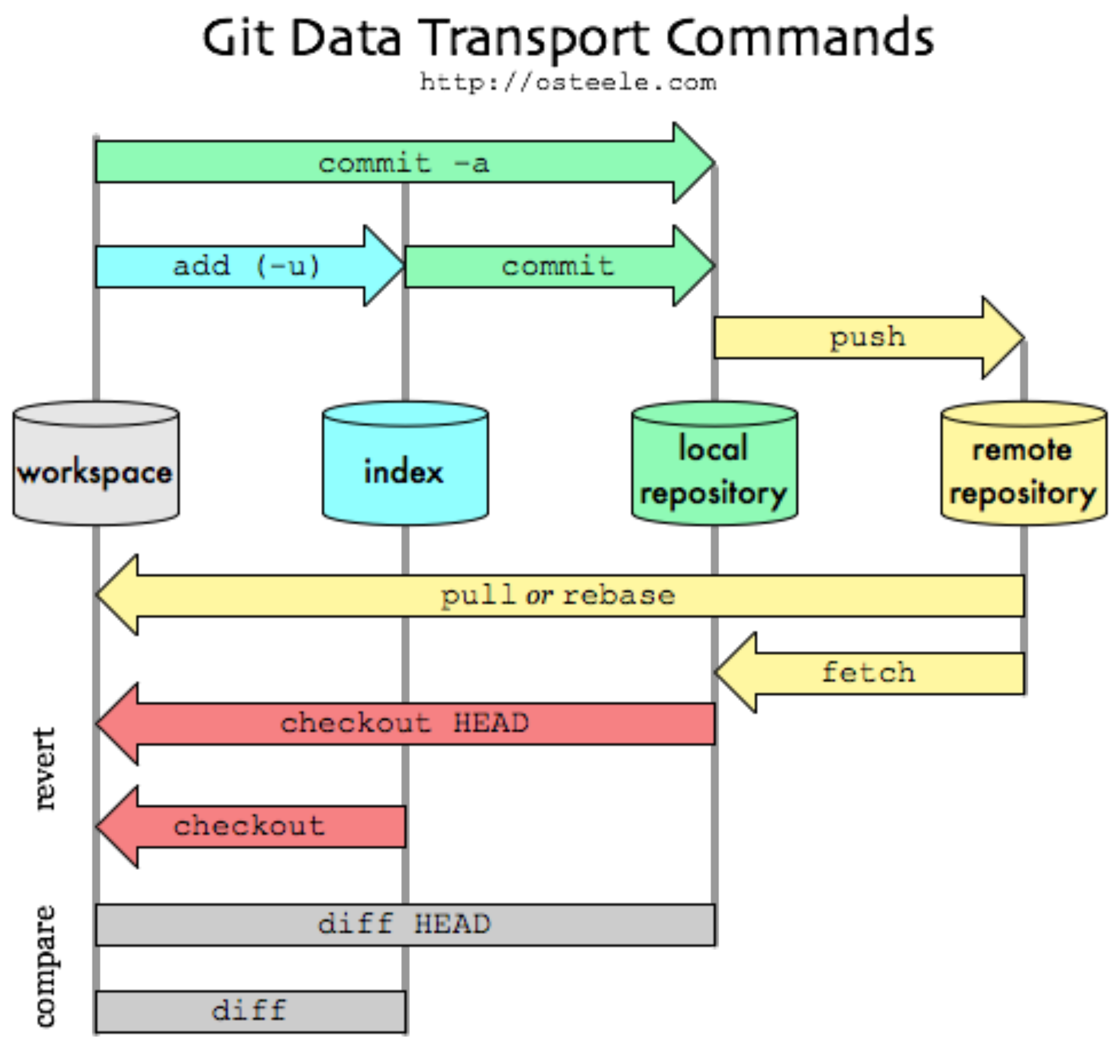


Remote repositories



Remote repositories

- Cloning an existing repo:
    - `$ git clone <url>`
- Create a Github account

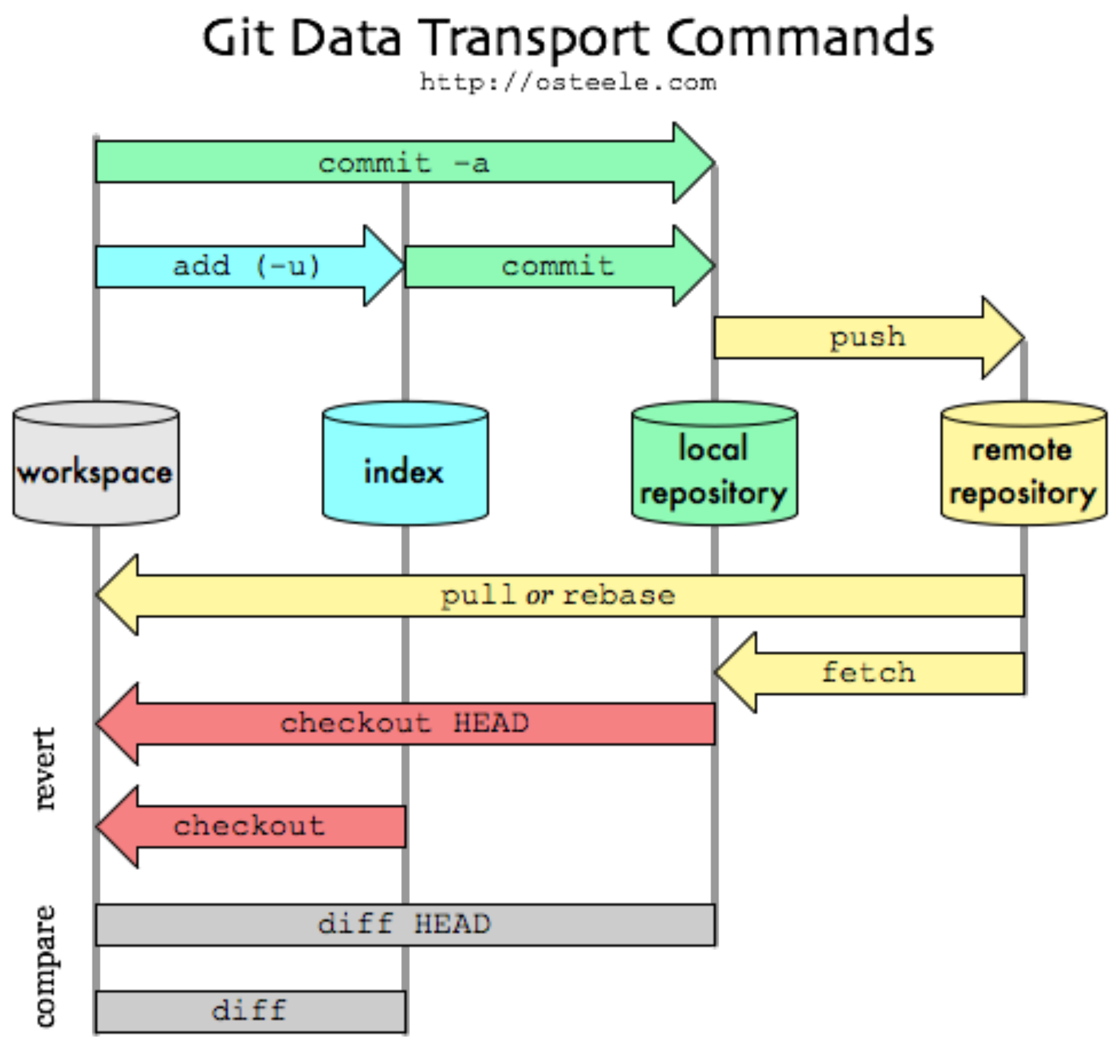


Remote repositories

- Check what remotes you have added:
  - `$ git remote -v`
- `Manually add new remote`
  - `$ git remote add <optional shortname> <url>`

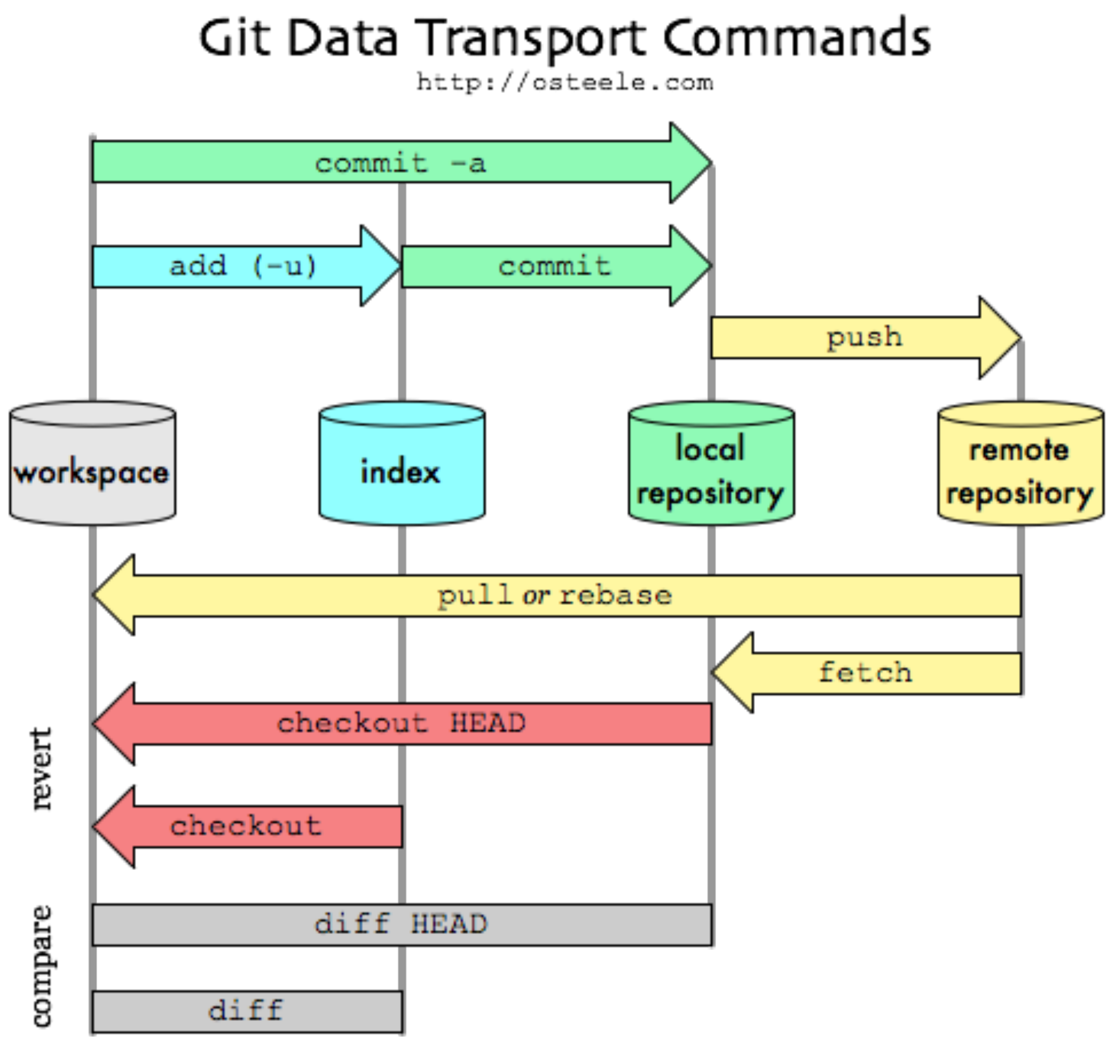
Remote repositories

- Get the data from the remote repo, add it to your local repo:
    - `$ git fetch`
- Get the data from the remote repo, and check it out
    - `$ git pull`

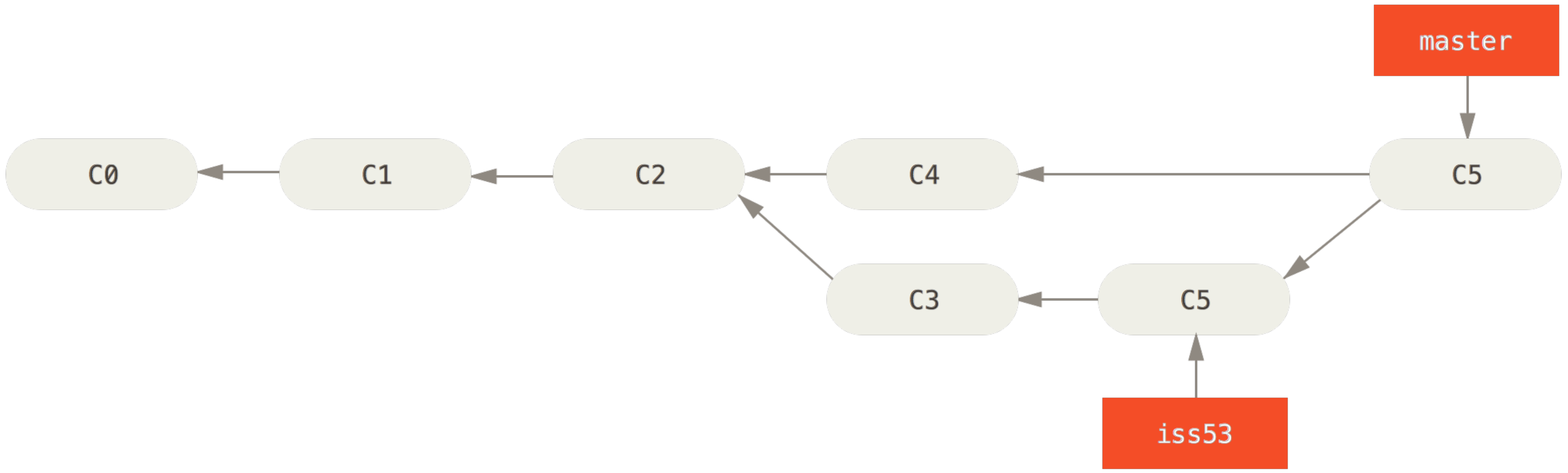


Remote repositories

- Send the files from your local repo to the remote repo:
  - `$ git push`

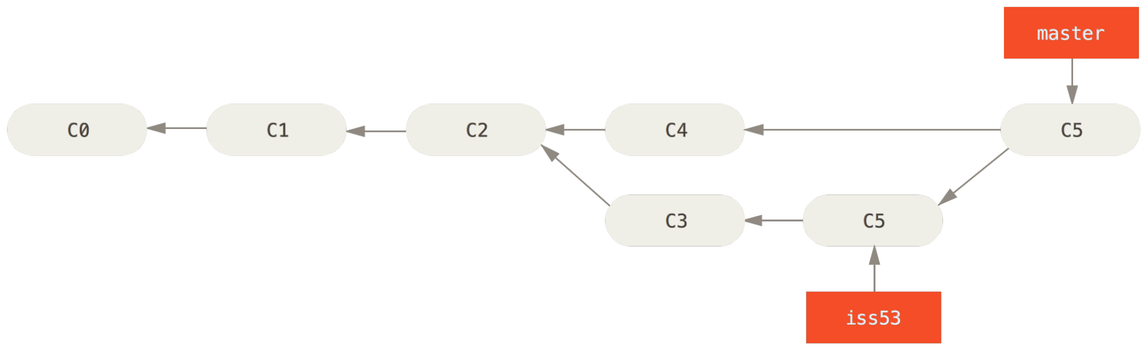


Branches



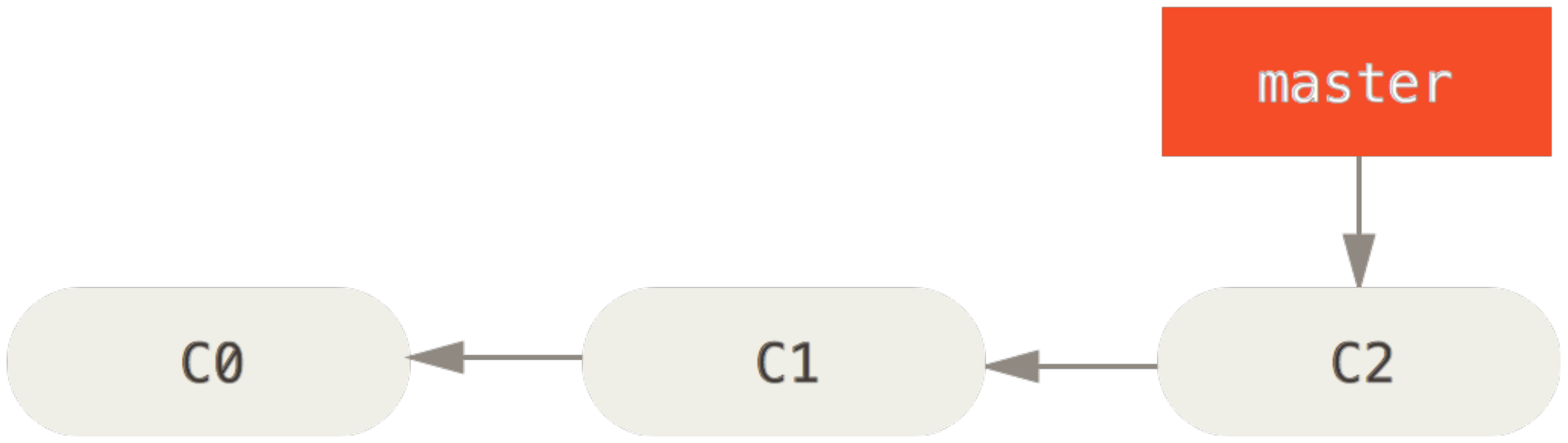
Branches

- Branches are pointers to a particular commit
- Allow you to work on parts of large projects individually
- Keep a stable working version of the code



Branches

- So far:

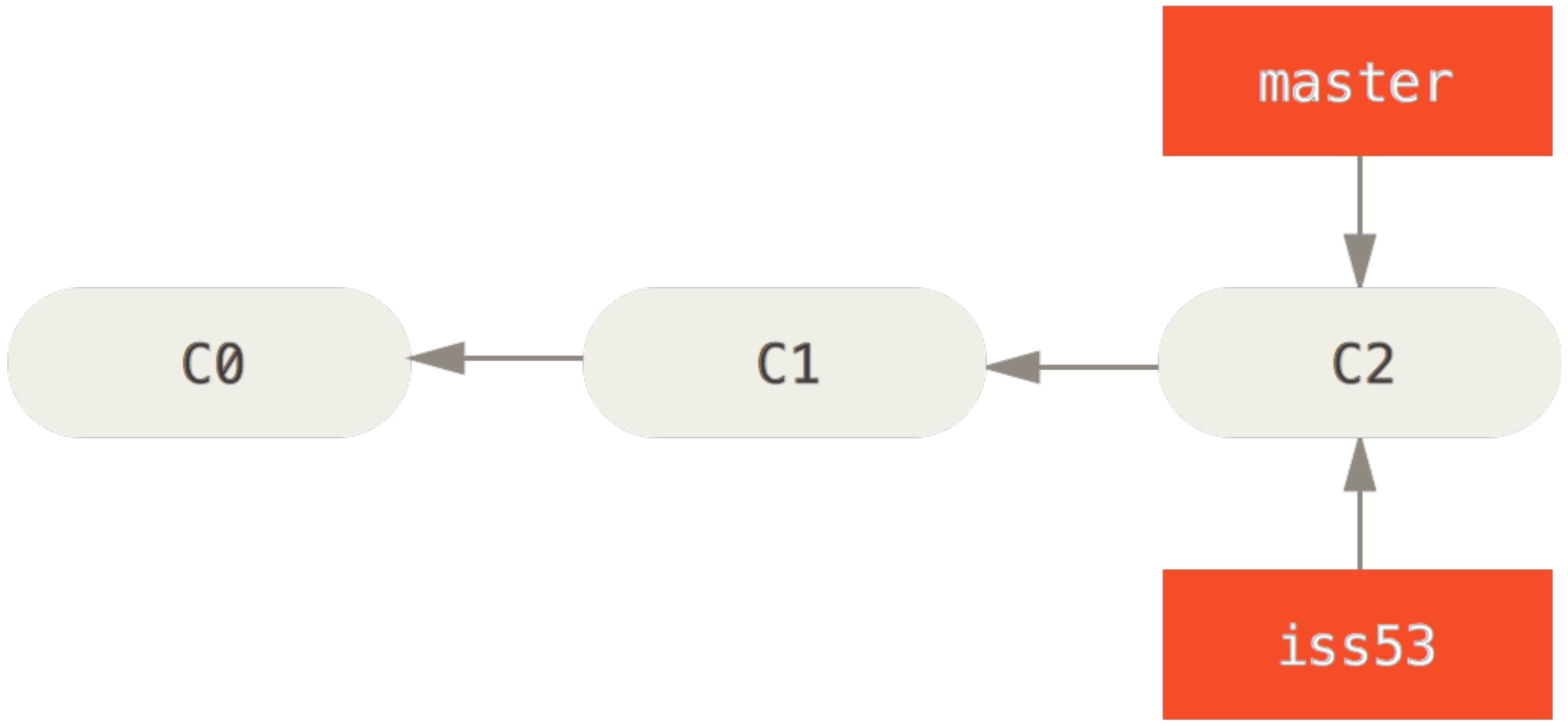


Branches

- Create a new branch:
  - `$ git branch <branch name>`
- Switch to the new branch
  - `$ git checkout <branch name>`
- Create & switch to new branch
  - `$ git checkout -b <branch name>`
- Wait, where am i?
  - `$ git branch`

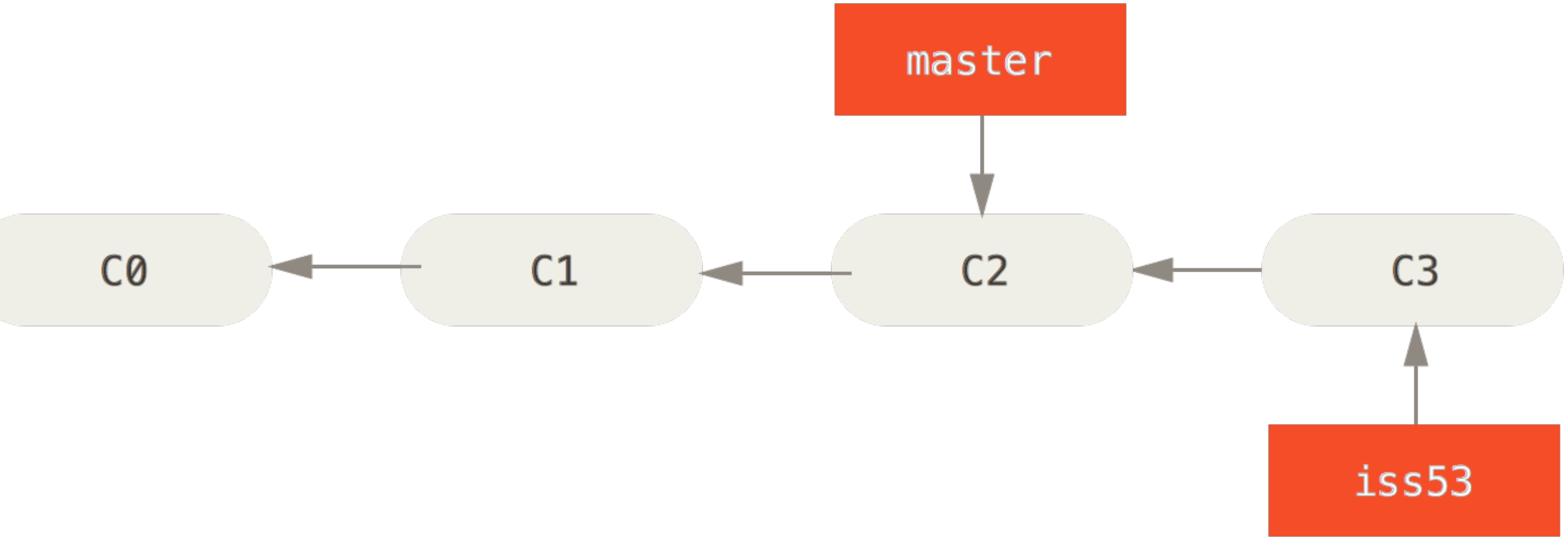
Branches

- Adding a branch does this:



Branches

- Doing a commit on a branch does this:

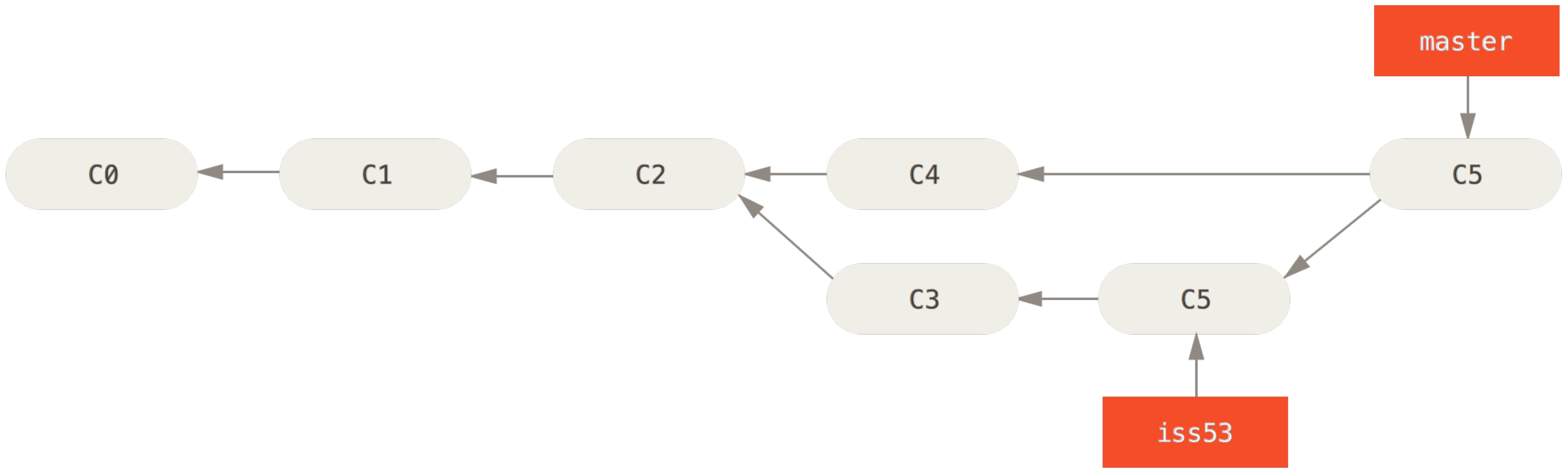


Branches

- Merging branches
  - `$ git checkout master`
  - `$ git merge <branch name>`
- If the merge is successful, and you are down with the branch
  - `$ git branch -d <branch name>`

Branches

- Changes from the branch are merged into the master branch

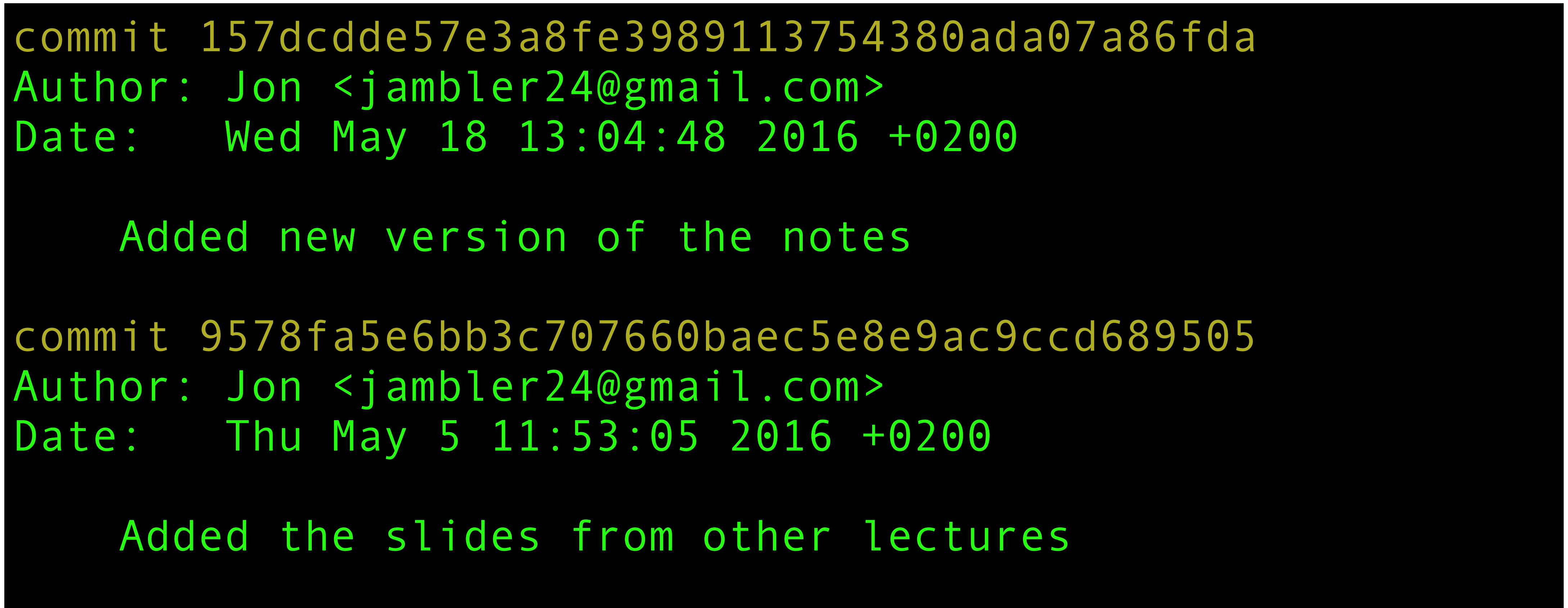


Retrieving old versions

First, find the correct version

- `git log`

This gives you all the commits made and the messages associated.



```
commit 157dcdde57e3a8fe3989113754380ada07a86fda
Author: Jon <jambler24@gmail.com>
Date:   Wed May 18 13:04:48 2016 +0200

    Added new version of the notes

commit 9578fa5e6bb3c707660baec5e8e9ac9ccd689505
Author: Jon <jambler24@gmail.com>
Date:   Thu May 5 11:53:05 2016 +0200

    Added the slides from other lectures
```

Retrieving old versions

First, find the correct version

- `git log`

This gives you all the commits made and the messages associated.

- `git checkout <commit hash>`
- `git add <Restored file>`
- `git commit -m “Fixed it!”`

Other materials

- Intro video
  - https://www.youtube.com/watch?v=Y9XZQO1n_7c