Git Workshop

Syncing, Branching, and Merging
Charles Guan

Topics

- Sync changes across different computers
- What's going on underneath?
- Collaborating on code
- Team workflows and extra tools

Why use version control at all?

- Back-up work
- Build off previous analyses
- Collaborate with teammates
- Regenerate figures 2 years from now
- Share code publicly (?)

To follow along:

- https://github.com/charlesincharge/tutorial_git/tree/tutorial-original
- http://learngitbranching.js.org

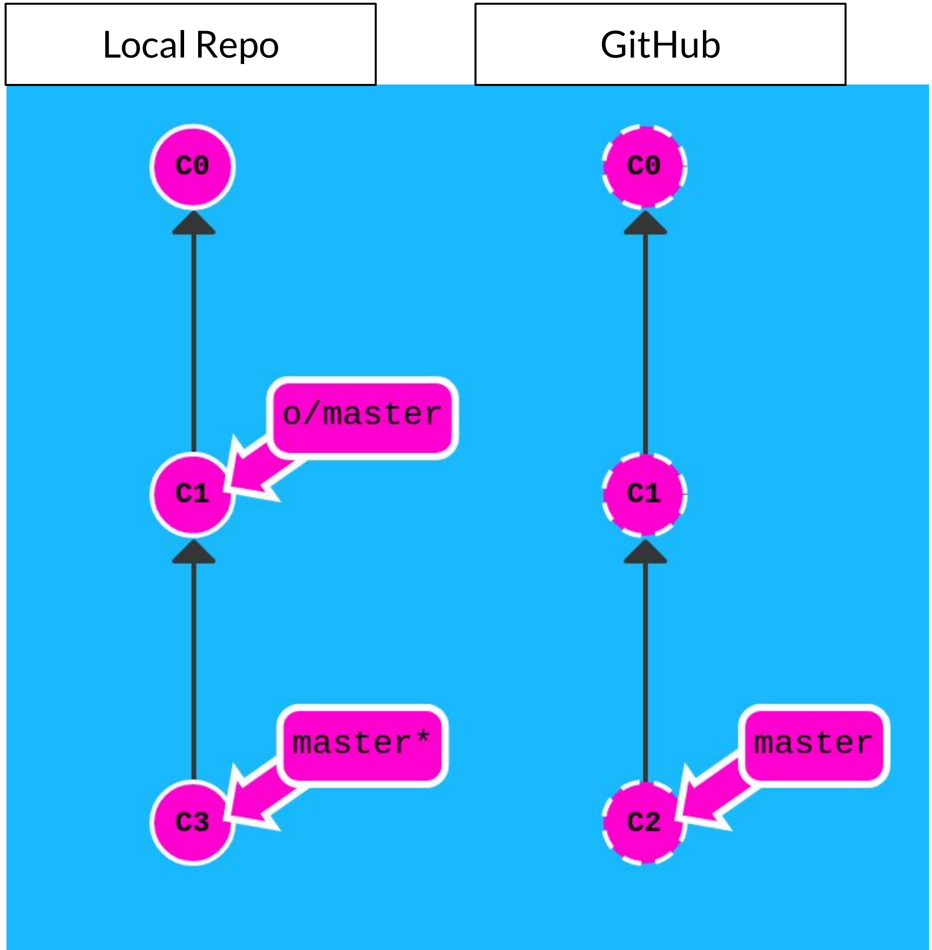
Synchronizing Changes

Pushing/Pulling

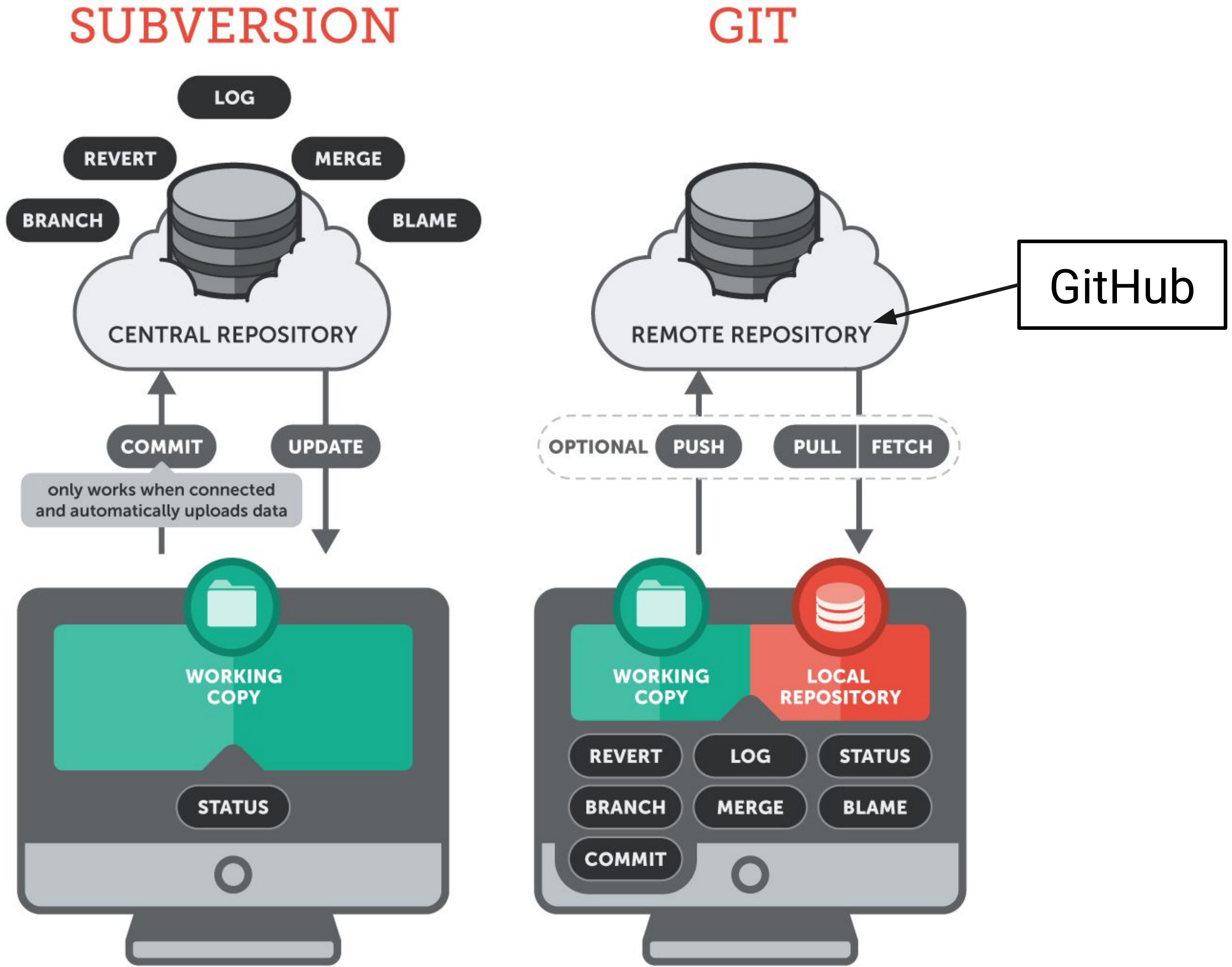
Pair up

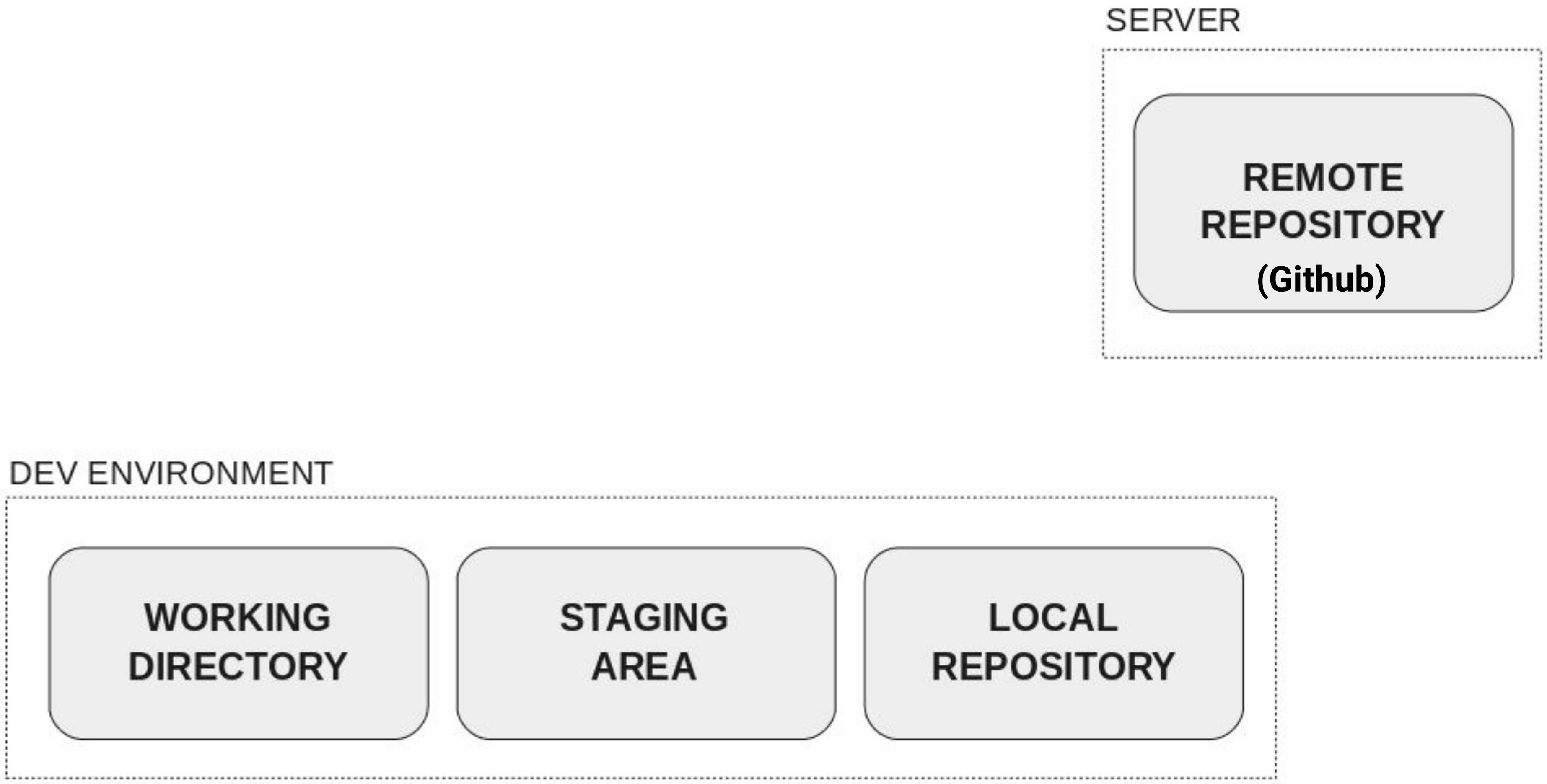
Open GitKraken or Git Bash

1. Create a new file on one computer
2. **Add** the file
3. **Commit** to local repository
4. **Push**
5. Pull on other computer

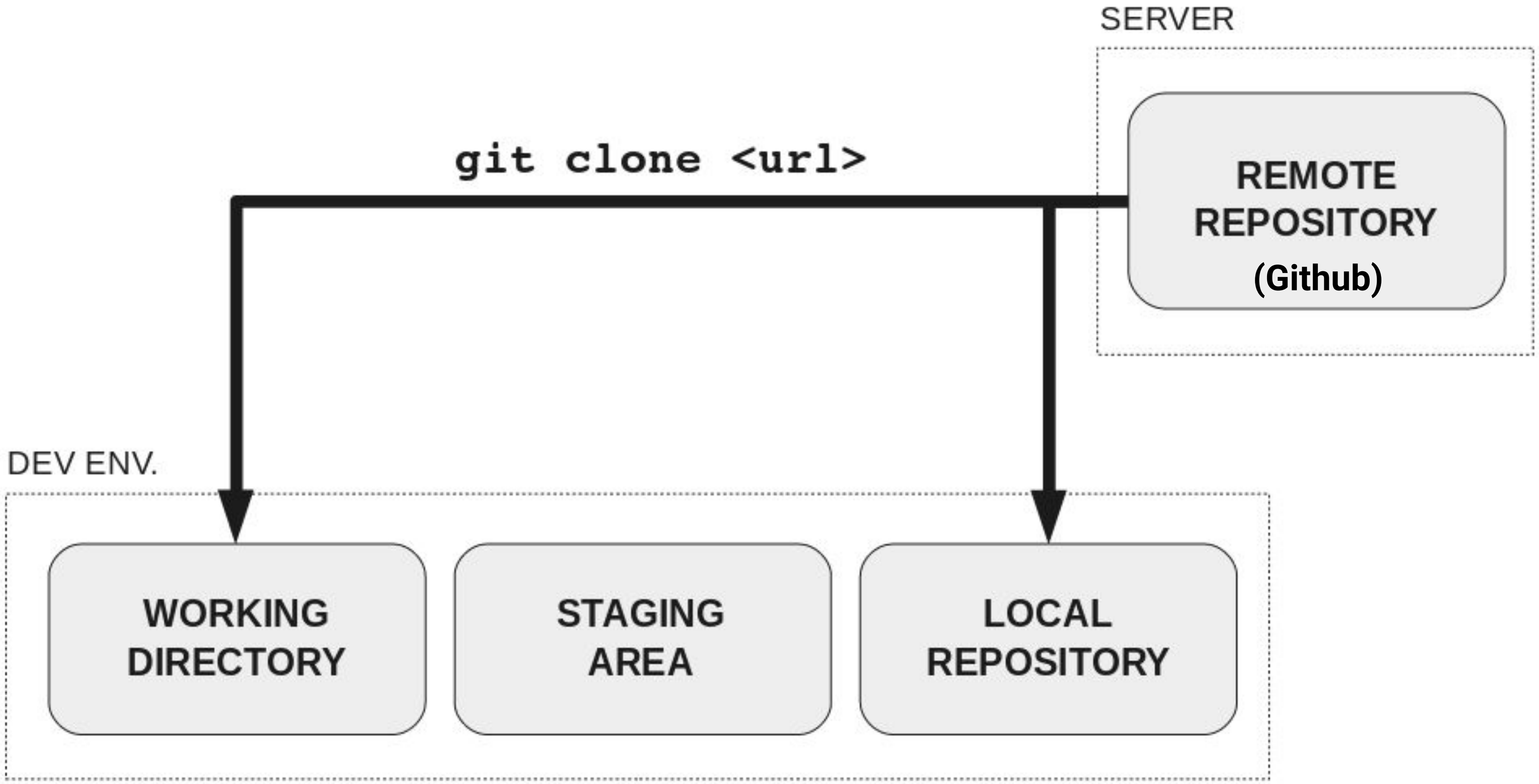


What's going on underneath?

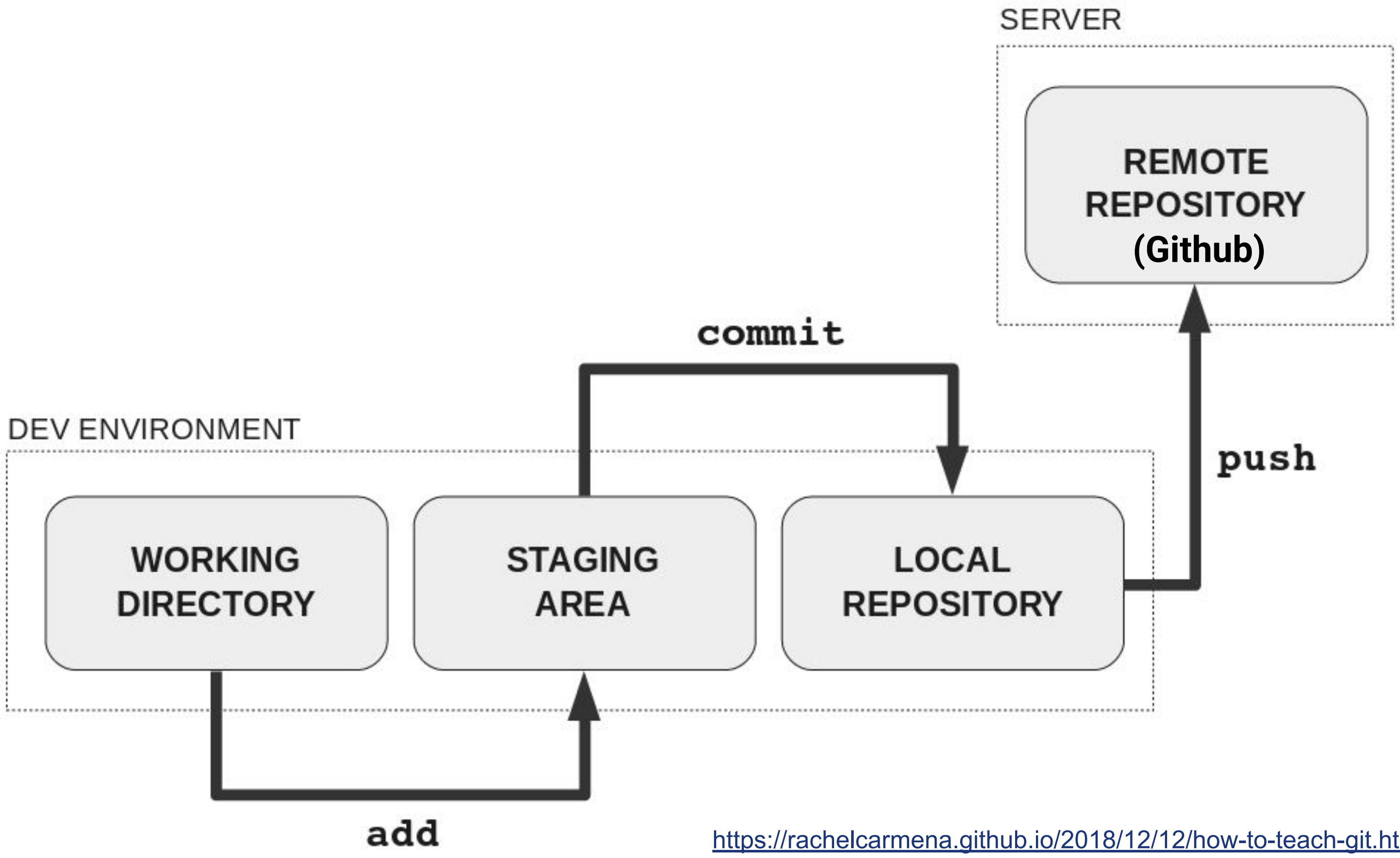

SUBVERSION
LOG
REVERT
MERGE
BRANCH
BLAME
CENTRAL REPOSITORY
COMMIT
UPDATE
only works when connected and automatically uploads data
WORKING COPY
STATUS
GIT
REMOTE REPOSITORY
GitHub
OPTIONAL
PUSH
PULL
FETCH
WORKING COPY
LOCAL REPOSITORY
REVERT
LOG
STATUS
BRANCH
MERGE
BLAME
COMMIT



https://rachelcarmena.github.io/2018/12/12/how-to-teach-git.html

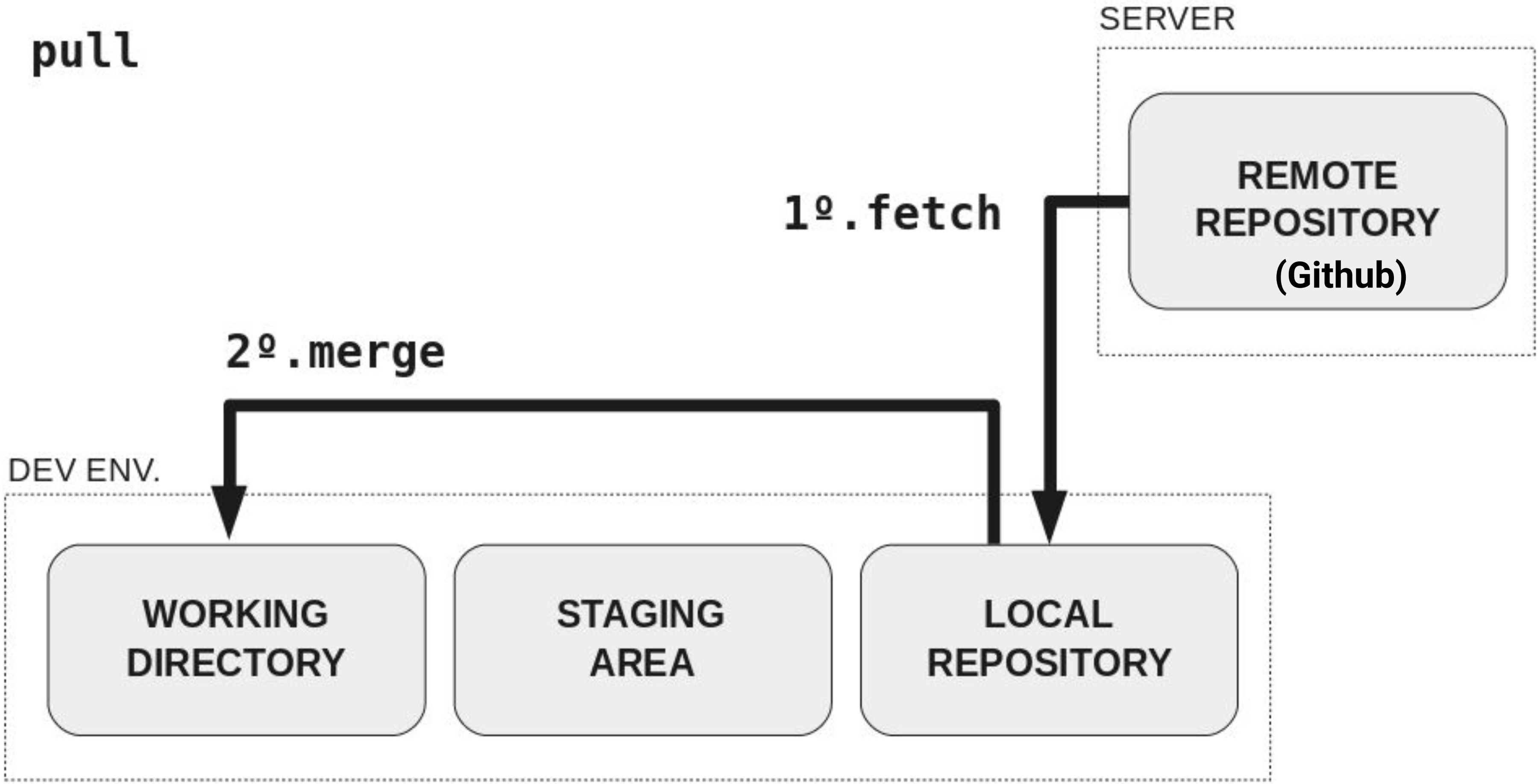


https://rachelcarmena.github.io/2018/12/12/how-to-teach-git.html



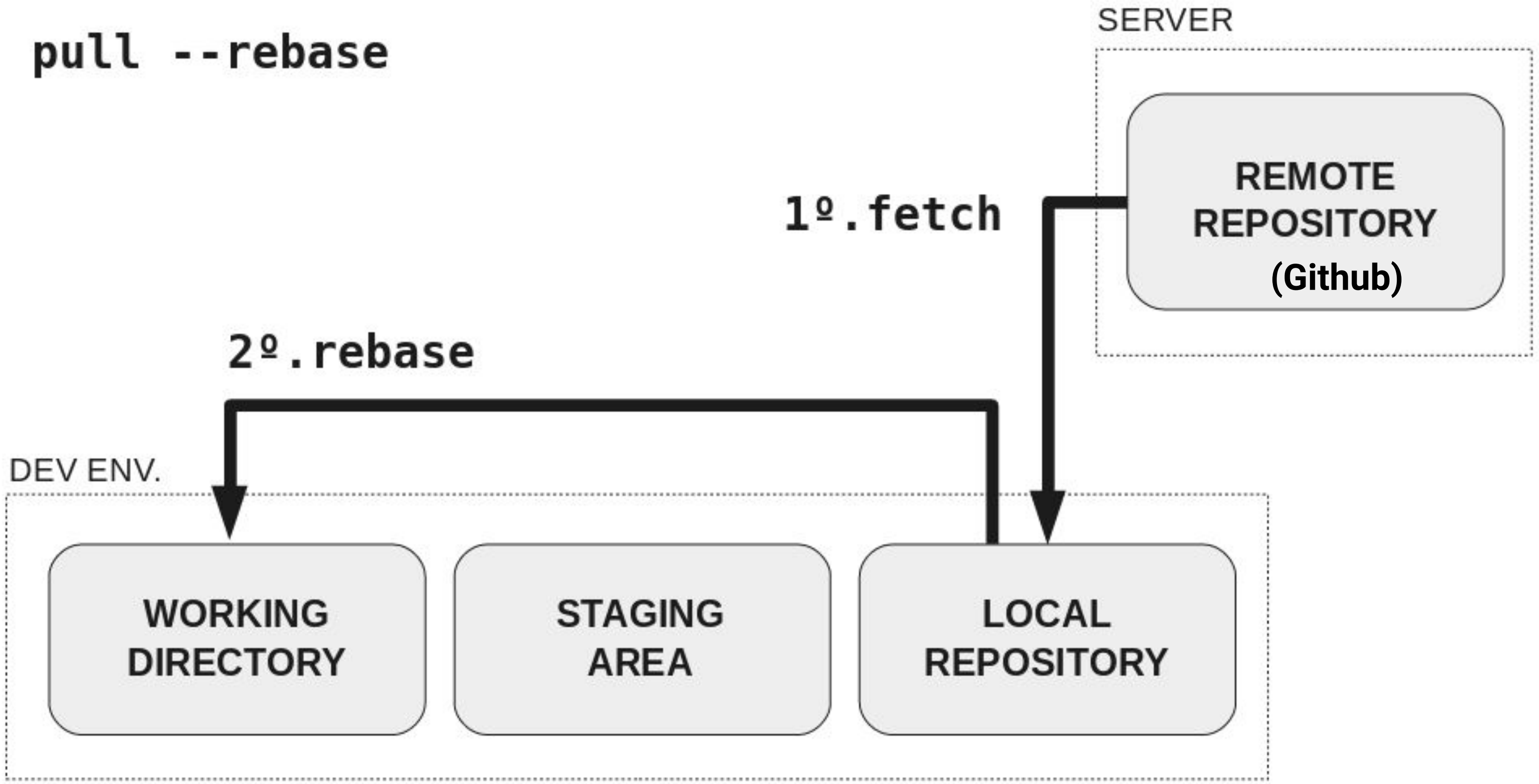
https://rachelcarmena.github.io/2018/12/12/how-to-teach-git.html

pull



https://rachelcarmena.github.io/2018/12/12/how-to-teach-git.html

`pull --rebase`



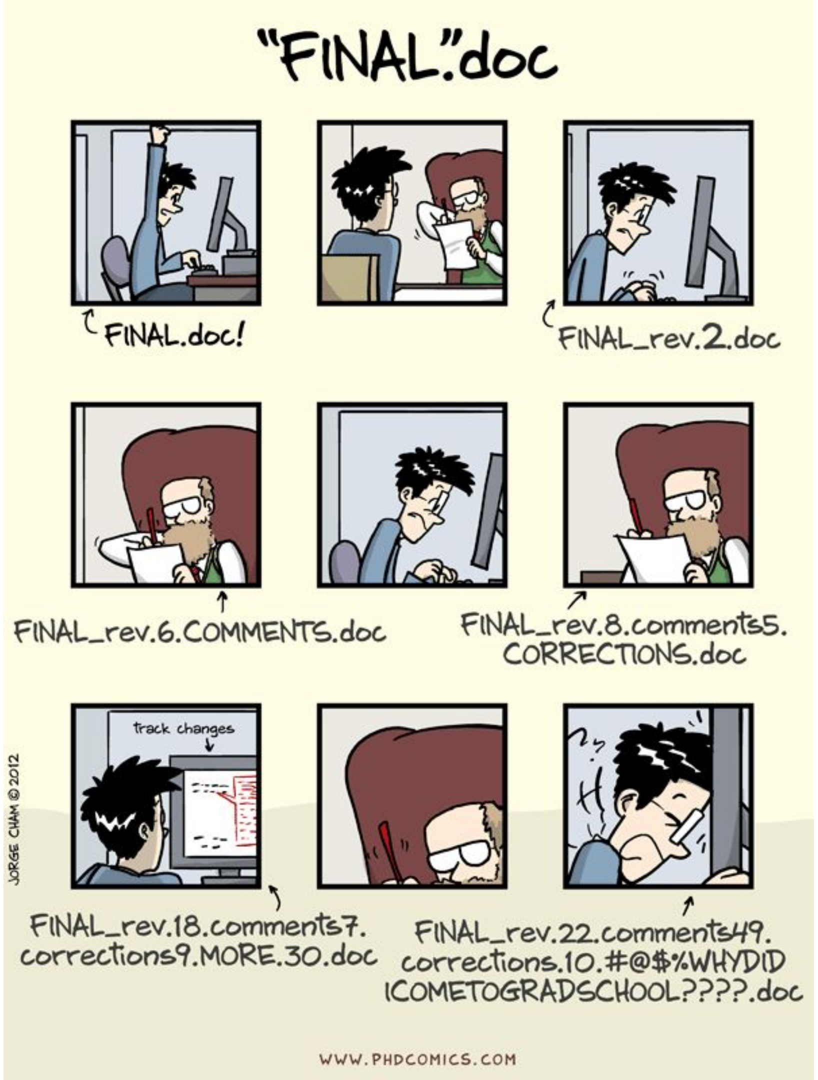
https://rachelcarmena.github.io/2018/12/12/how-to-teach-git.html

Merge Conflicts

Pair up

1. On both computers: modify the same line of the file you created earlier
2. Commit
3. Push/Pull
4. Resolve merge conflict
5. Push

Collaborating using Branches


"FINAL".doc
FINAL.doc!
FINAL_rev.2.doc
FINAL_rev.6.COMMENTS.doc
FINAL_rev.8.comments5.
CORRECTIONS.doc
track changes
FINAL_rev.18.comments7.
corrections9.MORE.30.doc
FINAL_rev.22.comments49.
corrections.10.#@$%WHYDID
ICOMETOGRADSCHOOL????.doc
JORGE CHAM © 2012
WWW.PHDCOMICS.COM

Branching / Merging

1. Create a new branch
2. Checkout branch
3. Add a file and commit the change
4. Push the branch to GitHub
5. Verify new branch is on GitHub
6. Merge your new branch into master
7. Push updated master

Cherry-pick-ing just 1 change

Pair up!

On 1st computer:

1. create a new branch
2. Fix buggy_rand.m
3. Push branch to GitHub

On 2nd computer

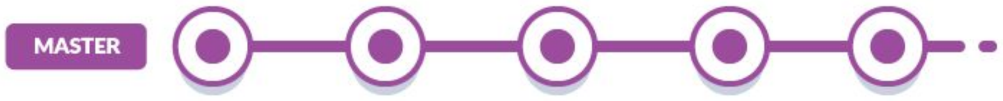
1. Fetch changes from GitHub
2. Cherry-pick fix for buggy_rand.m
3. (Optional) verify fix by running print_rand.m

Viewing branch history graph in GitHub/GitKraken

- https://github.com/charlesincharge/tutorial_git
- https://github.com/charlesincharge/tutorial_git/network

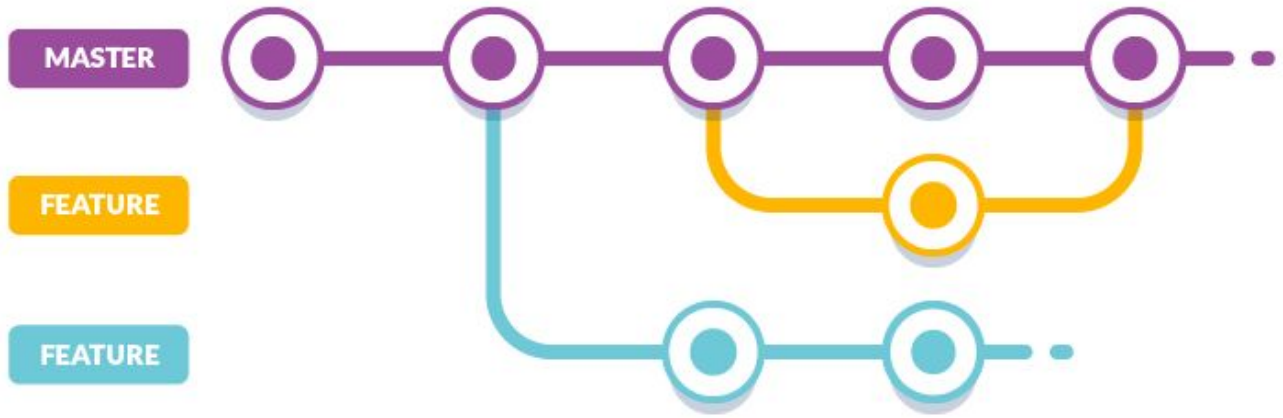
Git Workflows

Basic Workflow (what we do)



https://buddy.works/blog/5-types-of-git-workflows

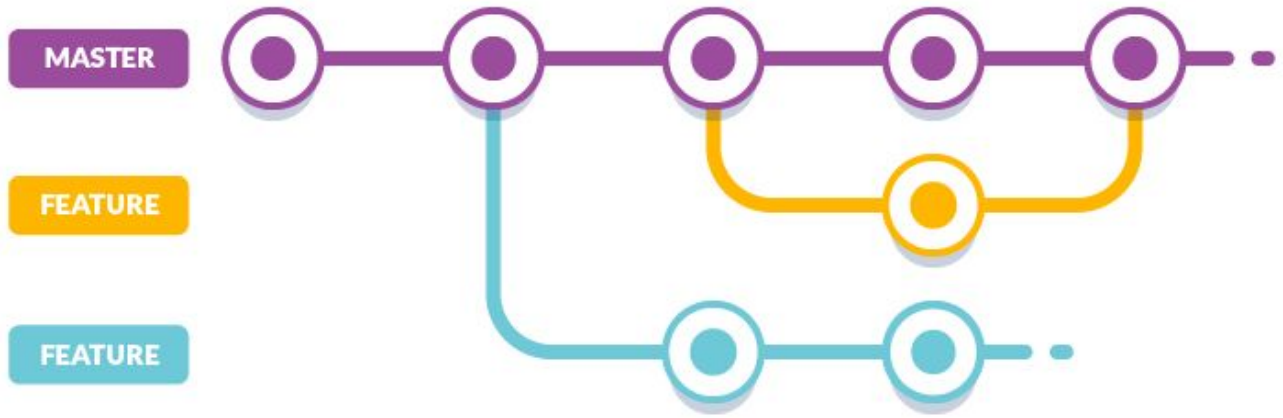
Feature Branching Workflow



https://buddy.works/blog/5-types-of-git-workflows

Feature Branching Workflow

"good"
branch



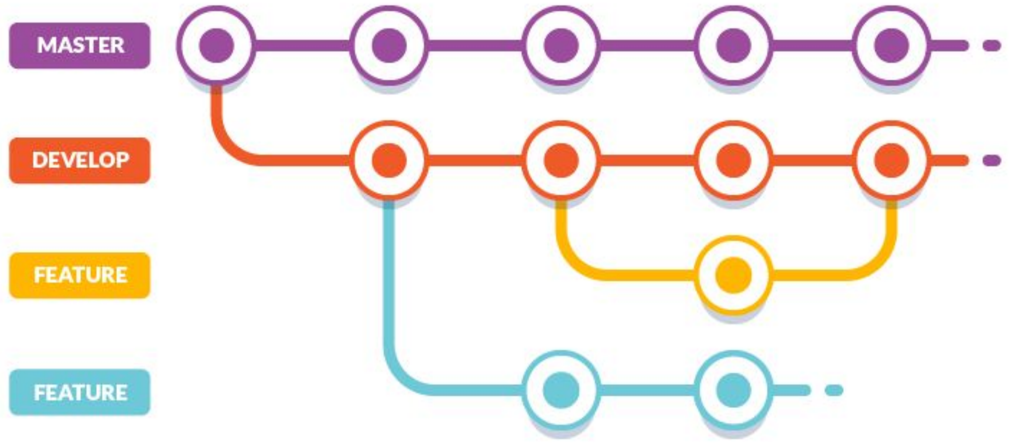
https://buddy.works/blog/5-types-of-git-workflows

GitFlow Workflow (huge projects)

tested and good to go

"probably-good" branch

work in progress



https://buddy.works/blog/5-types-of-git-workflows

Tips and Tricks

Tools

- git blame
- git bisect
- git tag
- git stash

What's the diff?

- merge vs rebase
- fetch vs pull vs pull --rebase
- revert vs reset

When to not use Git

- Large data files
  - .mat, .nev, .tif
- Formatted text files
  - .docx, .mlx

Thanks for listening!

Resources

- https://try.github.io/
- https://www.atlassian.com/git/tutorials