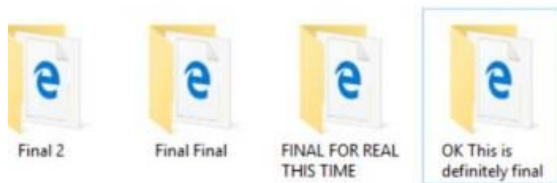


# C & N WORKSHOP: INTRO TO GIT/GITHUB

---

**Michelle Chiu**

michelle.chiu@temple.edu

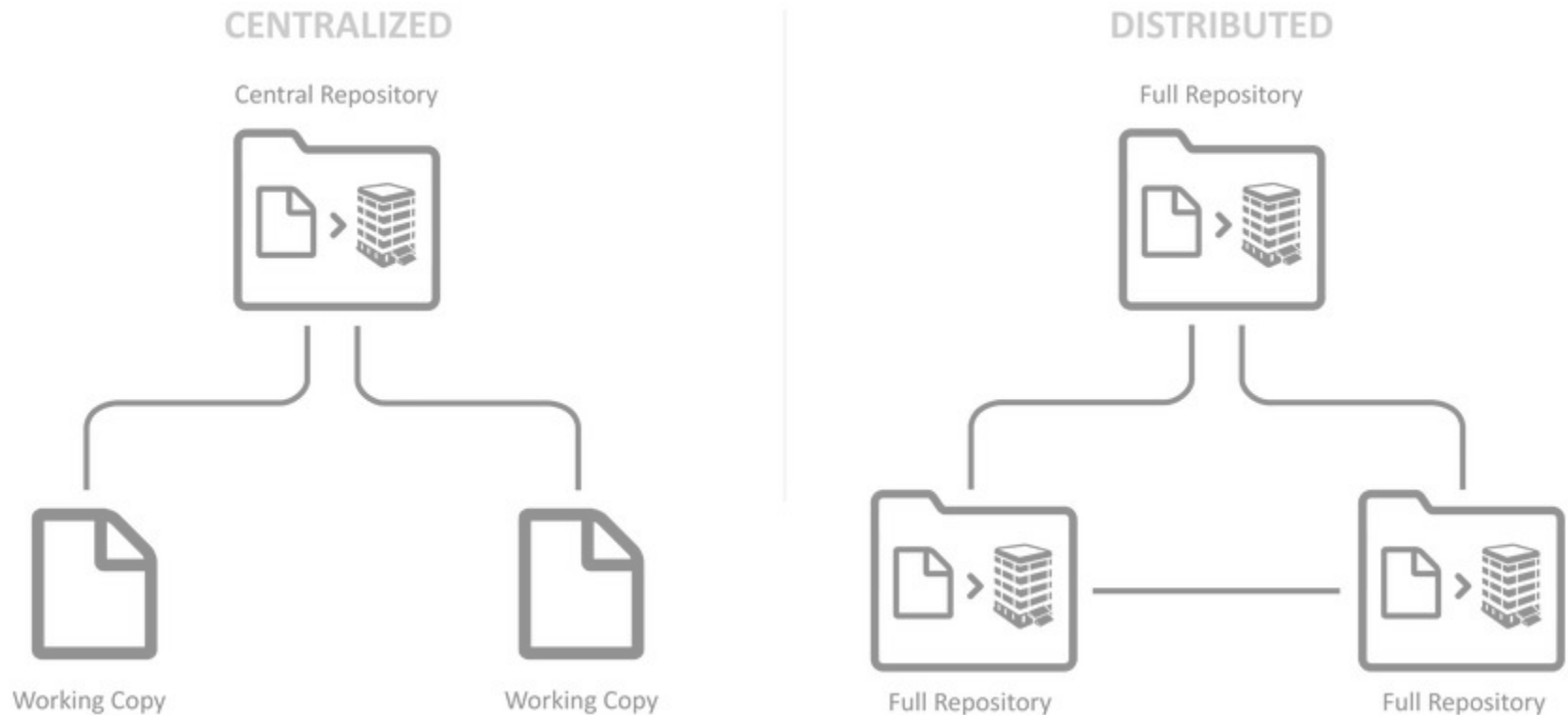


# What is Git?

**Repository (repo):** storage of files + their change history

# What is Git?

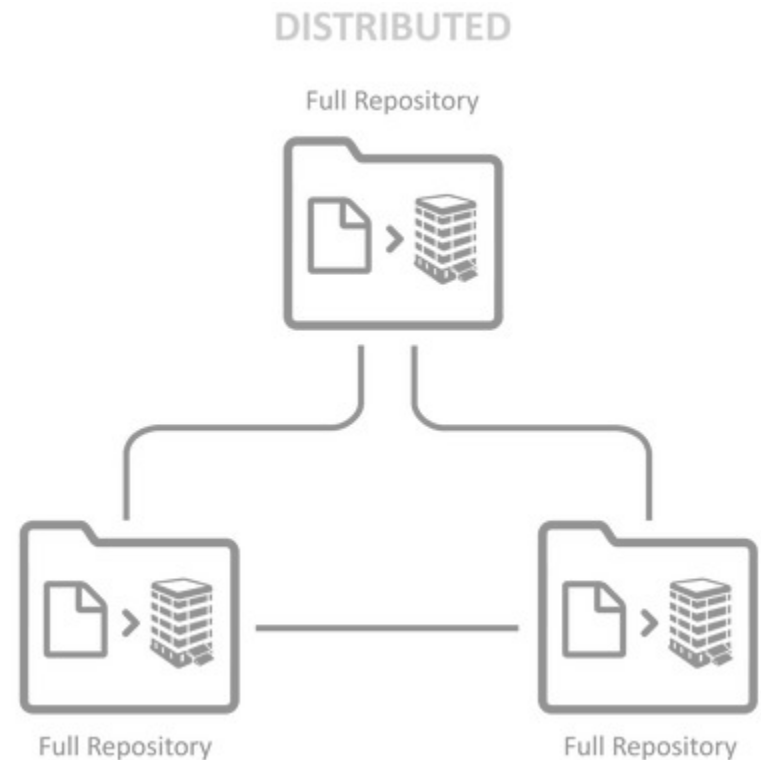
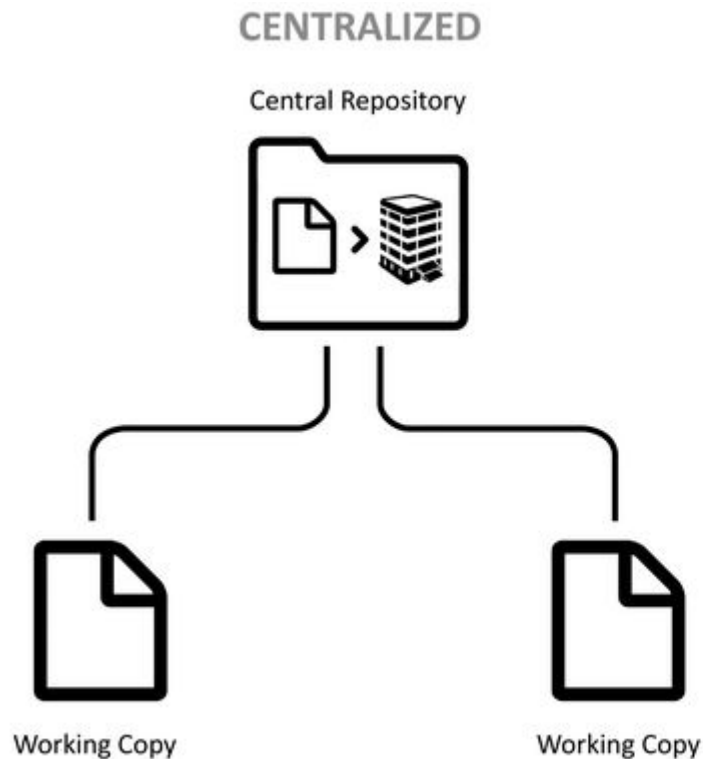
## 1) Distributed Version Control System (DVCS)



# What is Git?

## 1) Distributed Version Control System (DVCS)

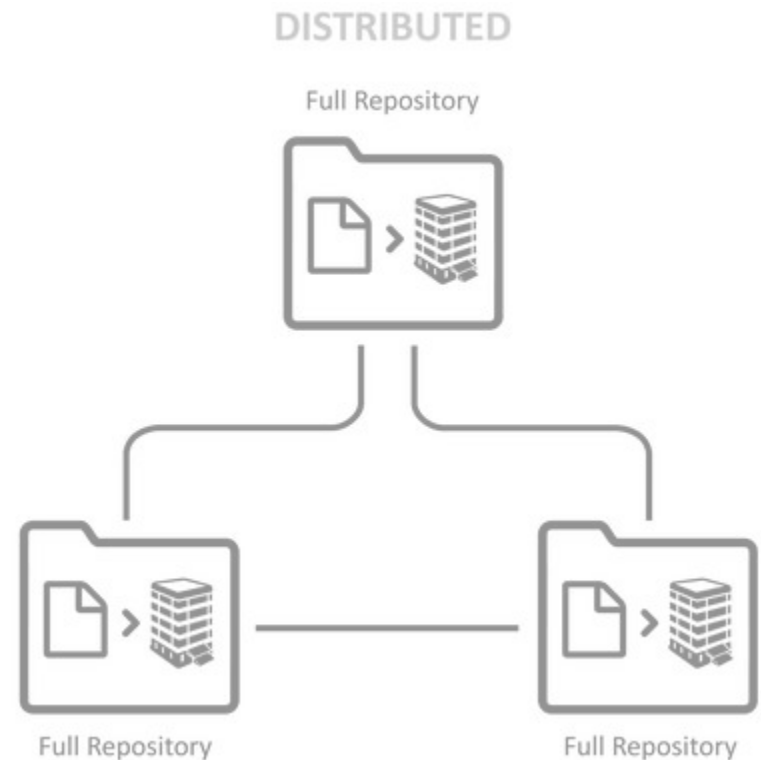
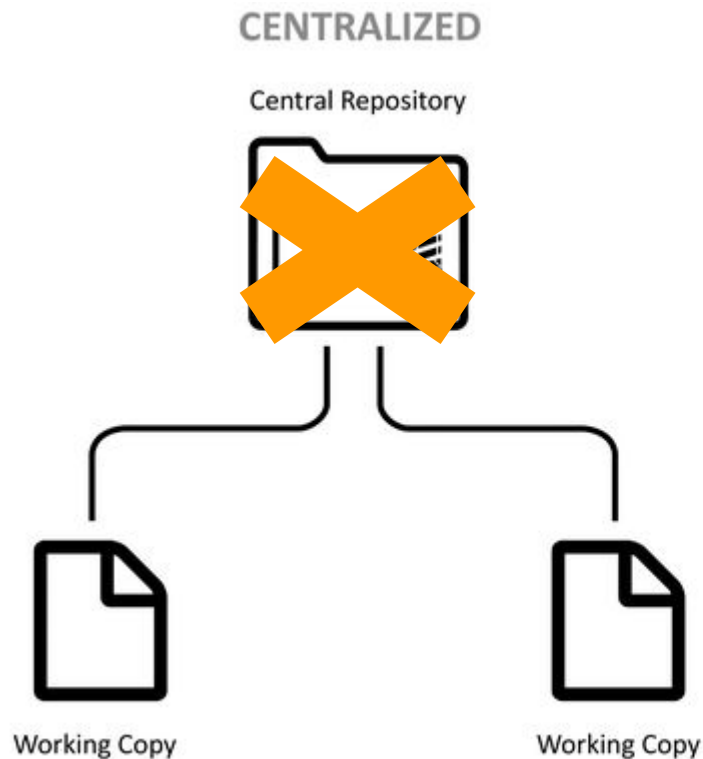
### client-server



# What is Git?

## 1) Distributed Version Control System (DVCS)

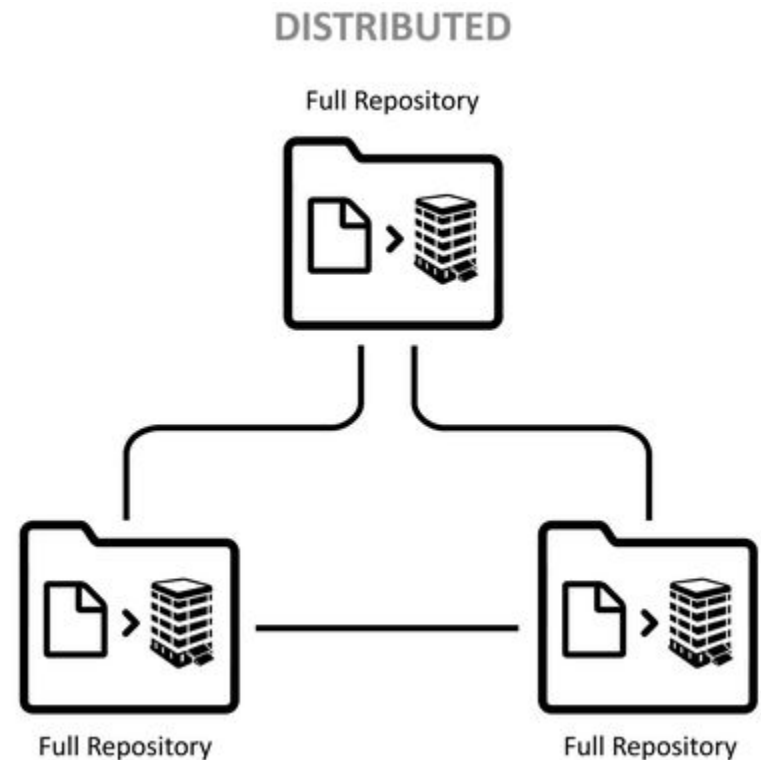
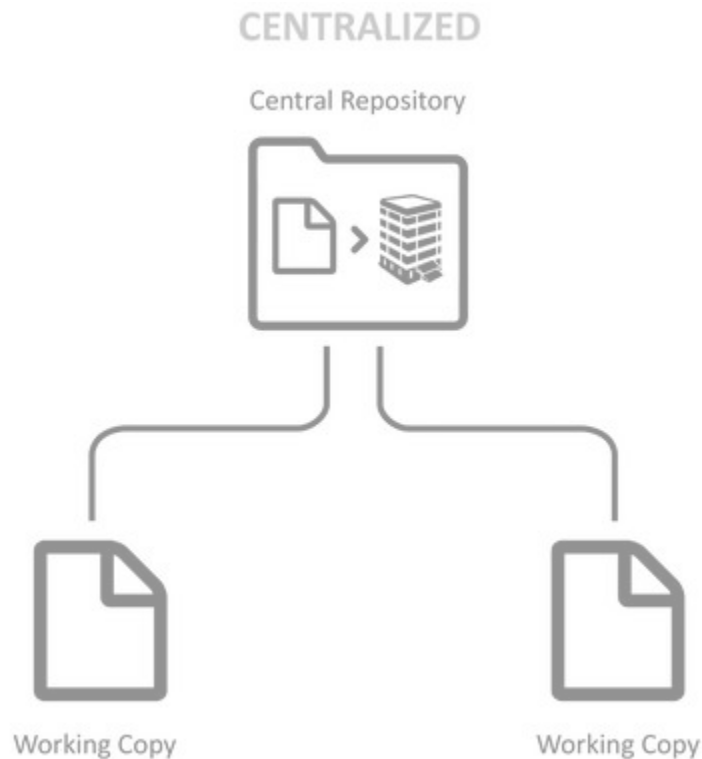
### client-server



# What is Git?

## 1) Distributed Version Control System (DVCS)

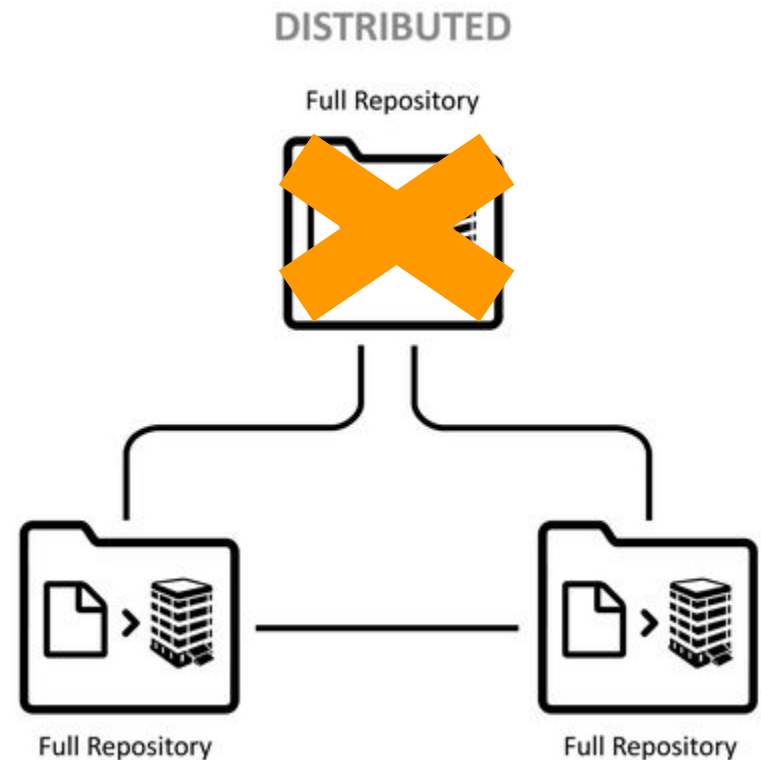
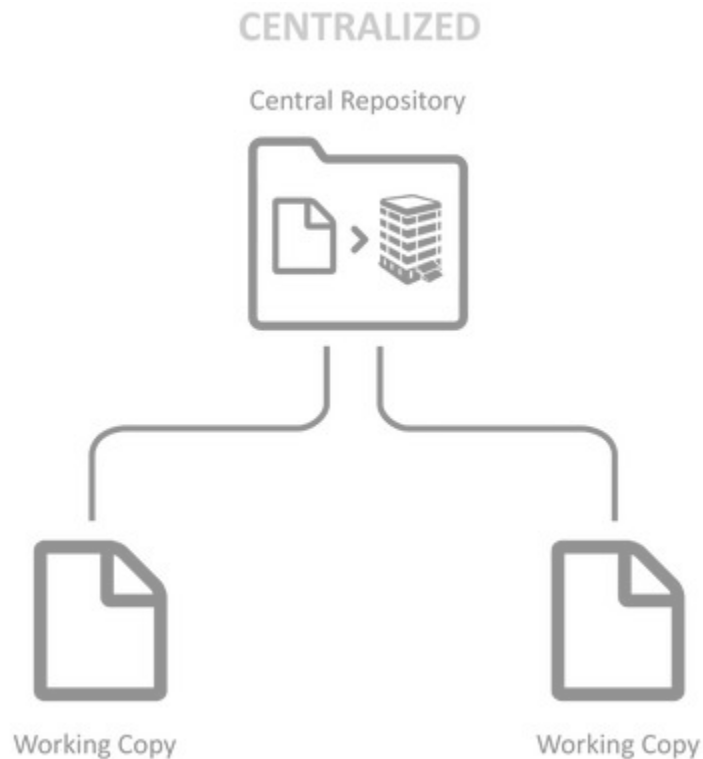
**peer-peer**



# What is Git?

## 1) Distributed Version Control System (DVCS)

**peer-peer**

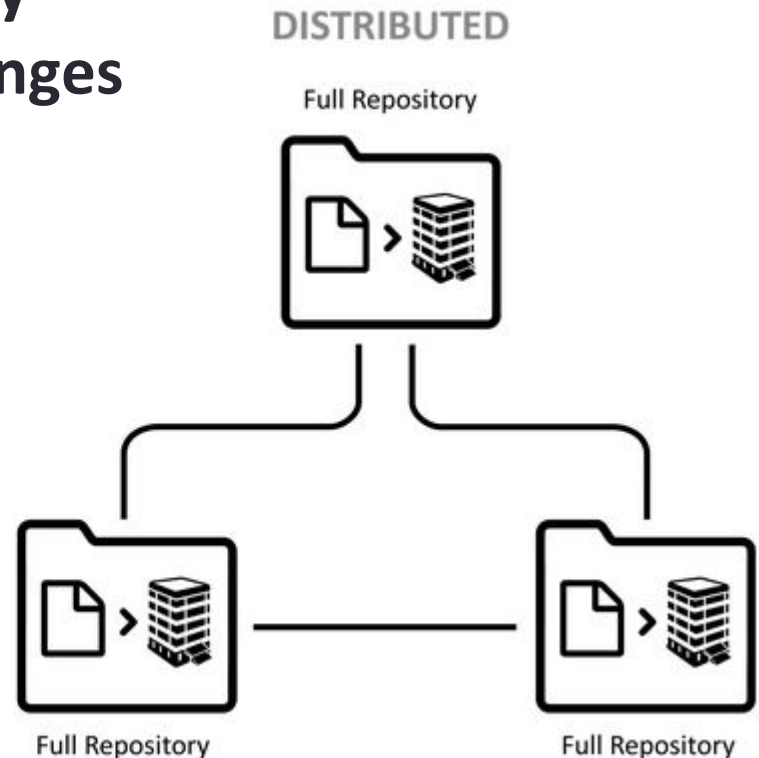




# What is Git?

## 1) Distributed Version Control System (DVCS)

- Every user has a **working copy** with **complete history of changes**
- Work locally **offline**



# Why Git?

- 2) **Granular control** over which **changes** are included in each **saved version**
- 3) Git rarely deletes anything



# Git and GitHub

**My Computer**

***LOCAL***

**GitHub**

***CLOUD***

# Exercise 1: Tell Git who you are



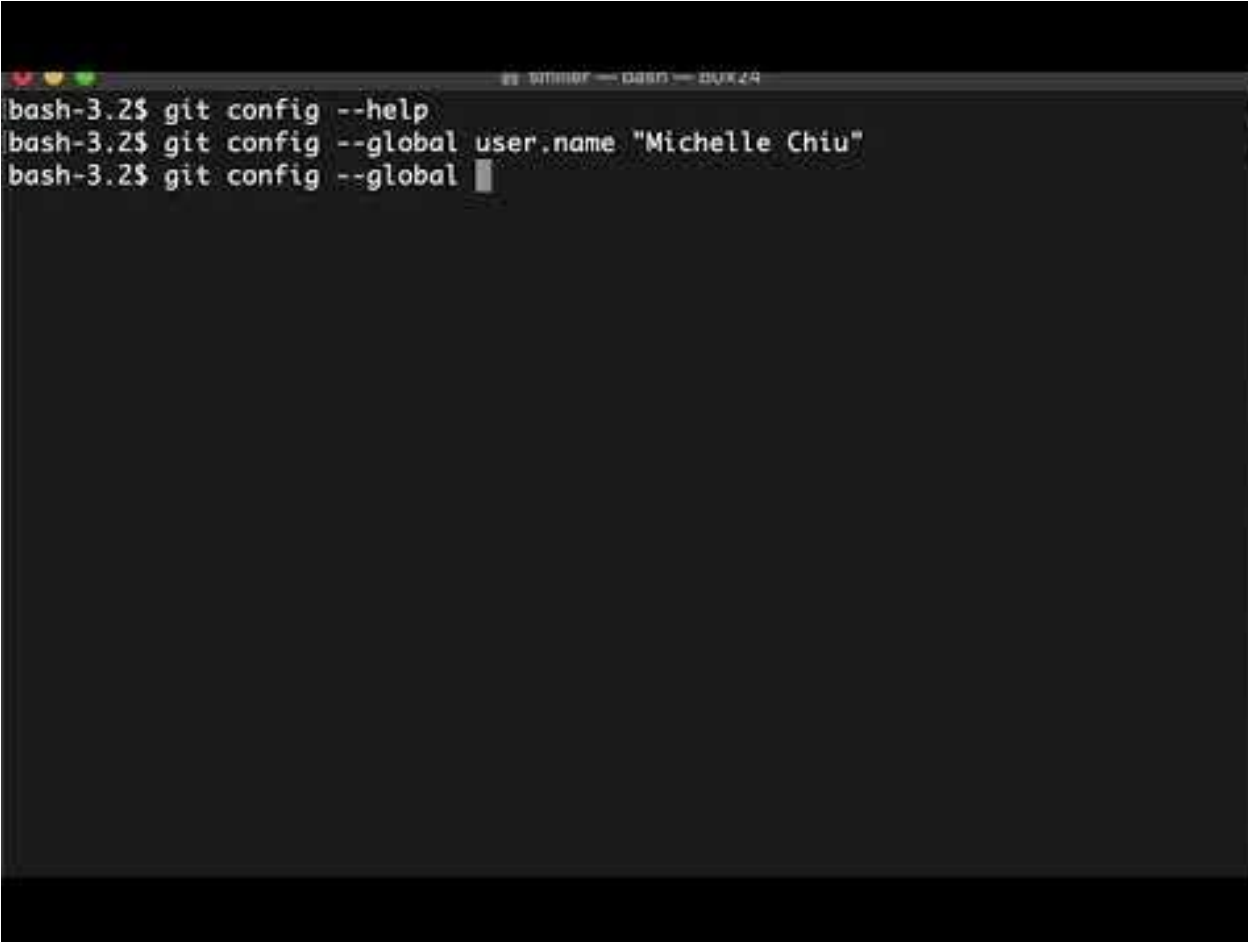
My Computer

*LOCAL*

Git records **WHO** and  
**WHAT** of changes

# Exercise 1: Tell Git who you are

[https://youtu.be/Sf\\_D7jAe-QI](https://youtu.be/Sf_D7jAe-QI)

A screenshot of a terminal window with a dark background. The window title bar shows standard macOS window controls (red, yellow, green buttons) and the text "SSH - 03/07/24 - 20:00:24". The terminal content shows three lines of commands entered in a bash shell:

```
bash-3.2$ git config --help
bash-3.2$ git config --global user.name "Michelle Chiu"
bash-3.2$ git config --global
```

The cursor is positioned at the end of the third line, ready for the next command.

# Exercise 1: Tell Git who you are

## My Computer

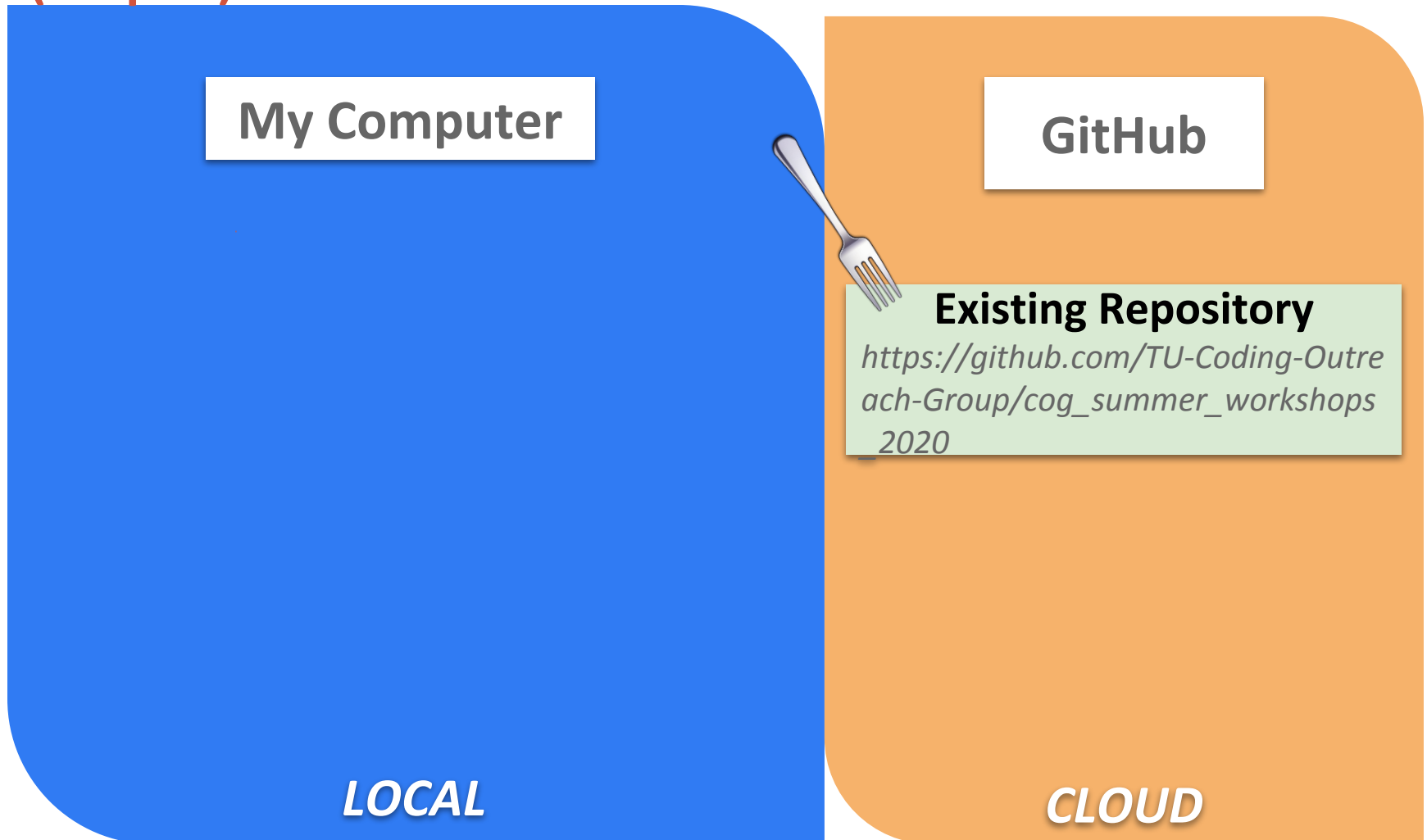
```
$ git config --global user.name "Your Name"  
$ git config --global user.email yourname@yourplace.org
```

```
$ cat ~/.gitconfig  
[user] name = Your Name email = yourname@yourplace.org  
[core] editor = nano
```

**LOCAL**

Git records **WHO** and **WHAT** of changes

# Working From An Existing Repository (repo)



# Working From An Existing Repository (repo)

My Computer

**LOCAL**

GitHub

## Existing Repository

[https://github.com/TU-Coding-Outreach-Group/cog\\_summer\\_workshops\\_2020](https://github.com/TU-Coding-Outreach-Group/cog_summer_workshops_2020)

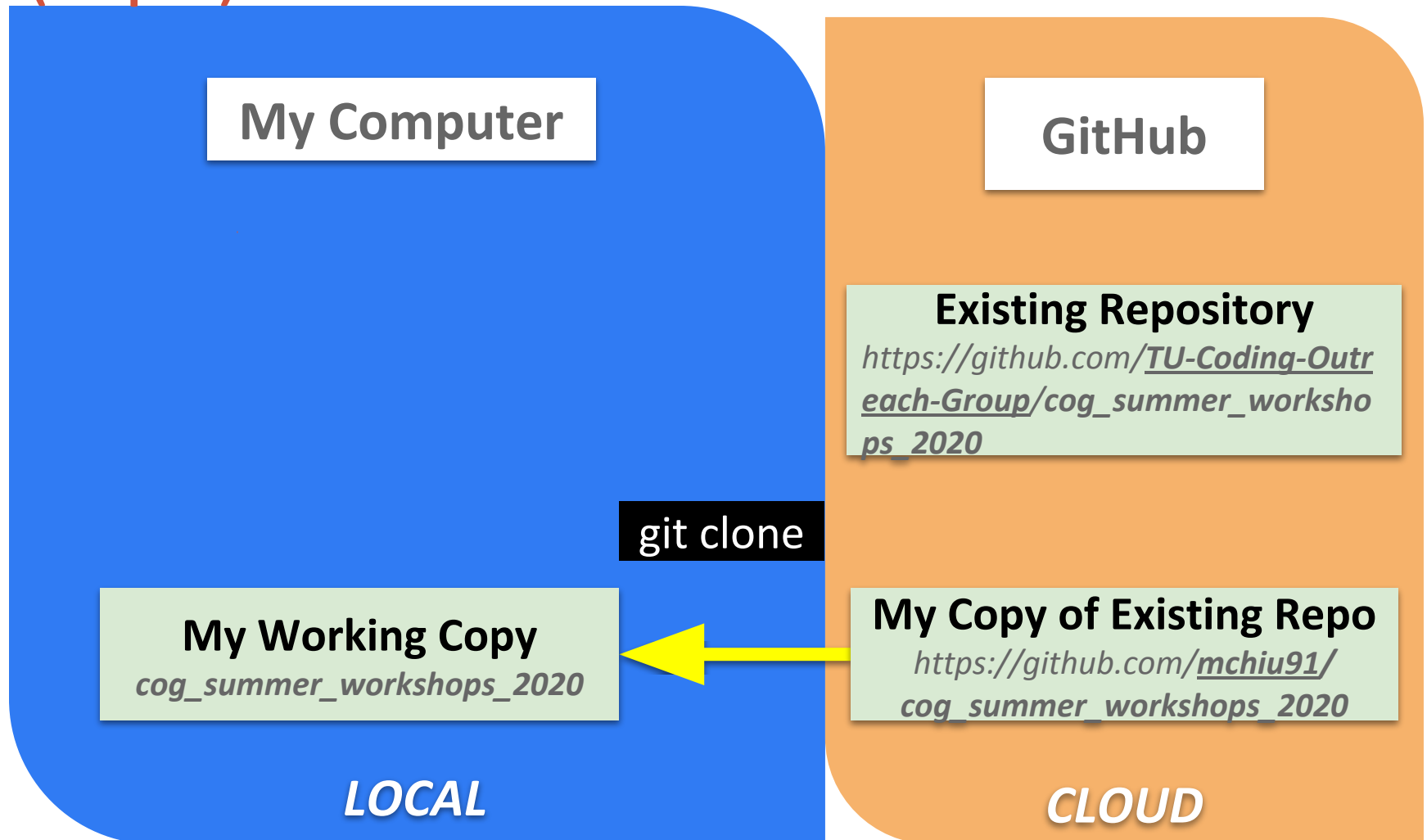
## My Copy of Existing Repo

[https://github.com/mchiu91/cog\\_summer\\_workshops\\_2020](https://github.com/mchiu91/cog_summer_workshops_2020)

**CLOUD**

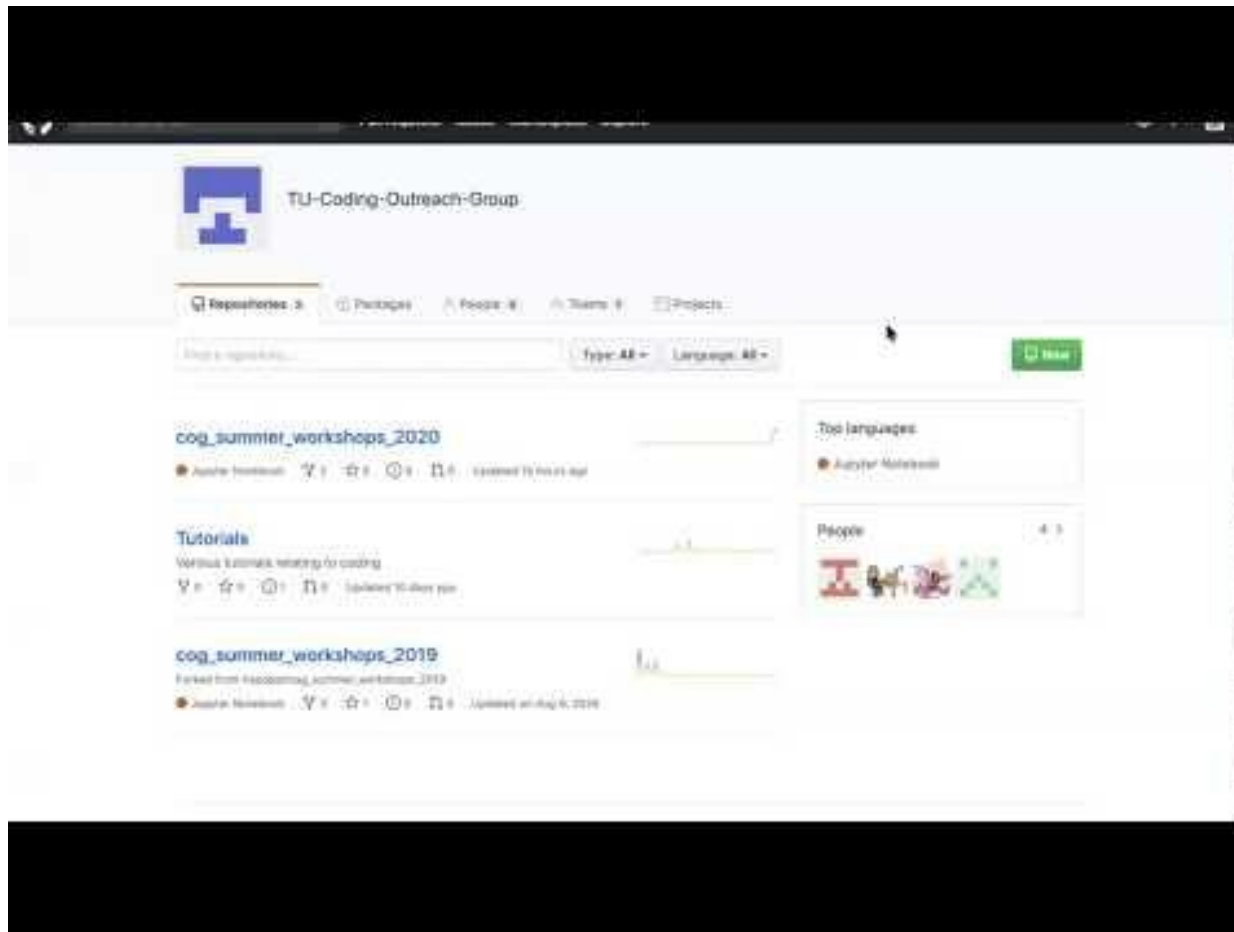


# Working From An Existing Repository (repo)

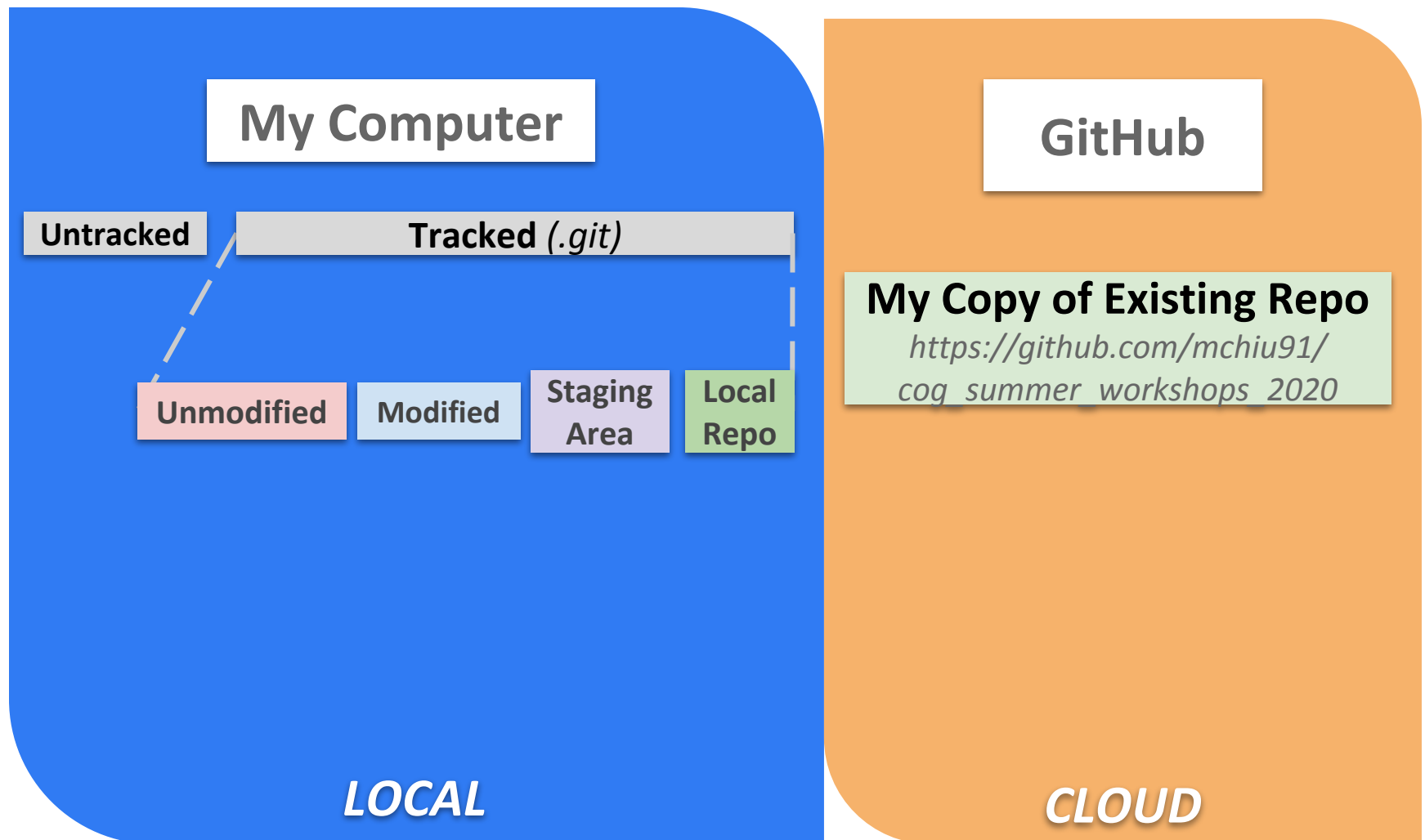


# Exercise 2: Fork and Clone an Existing Repository (repo)

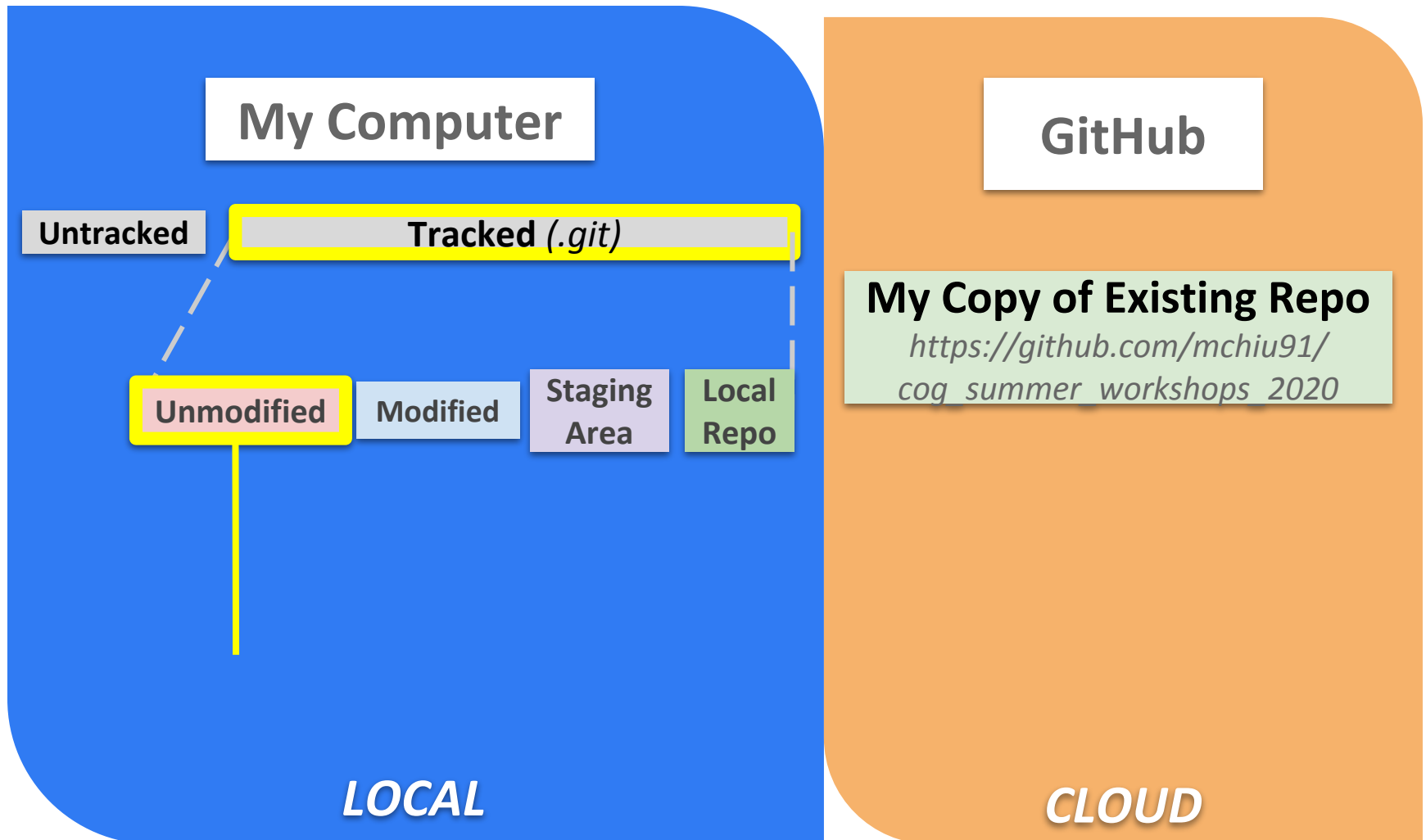
<https://youtu.be/zxdR95acSWs>



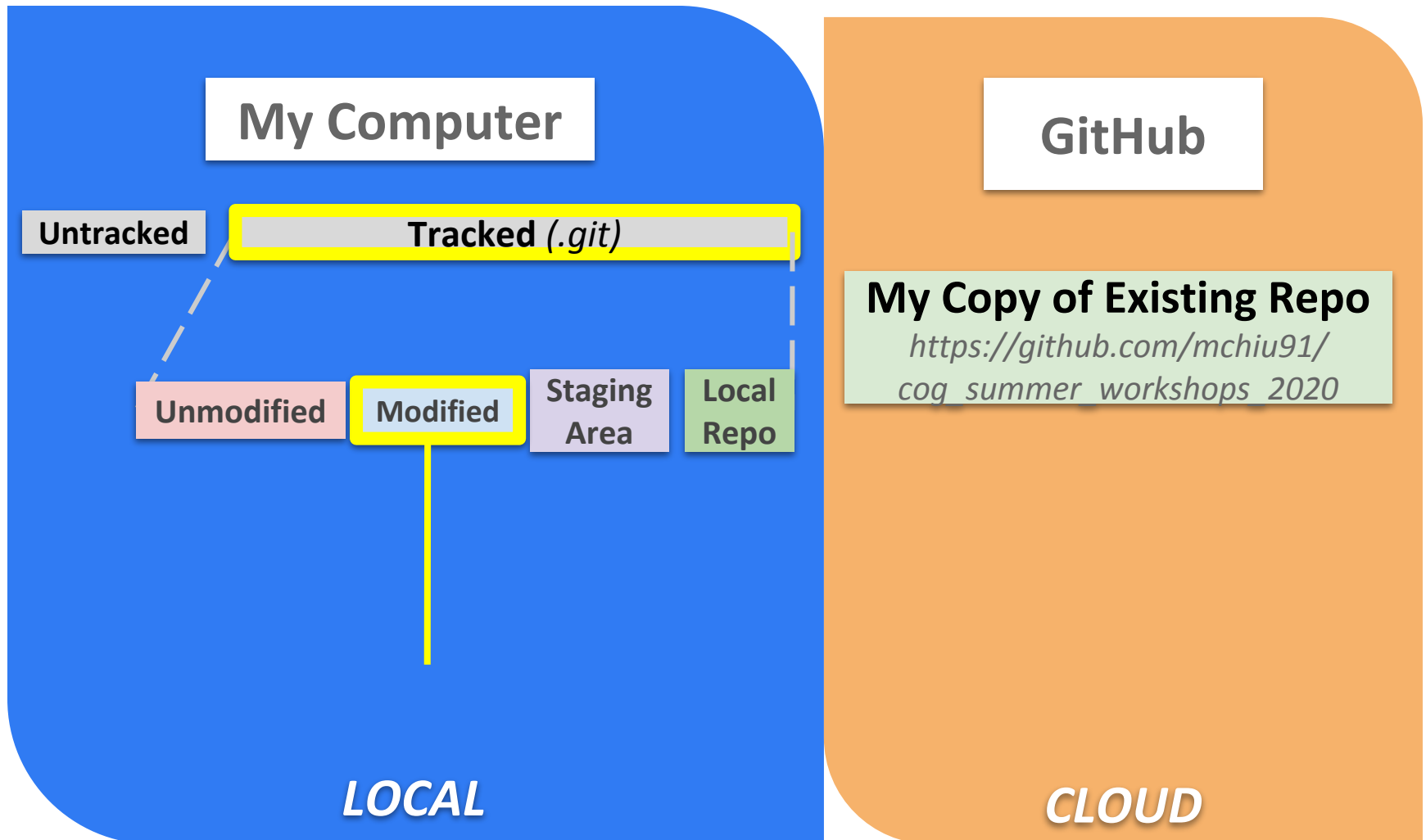
# Understanding Git File States



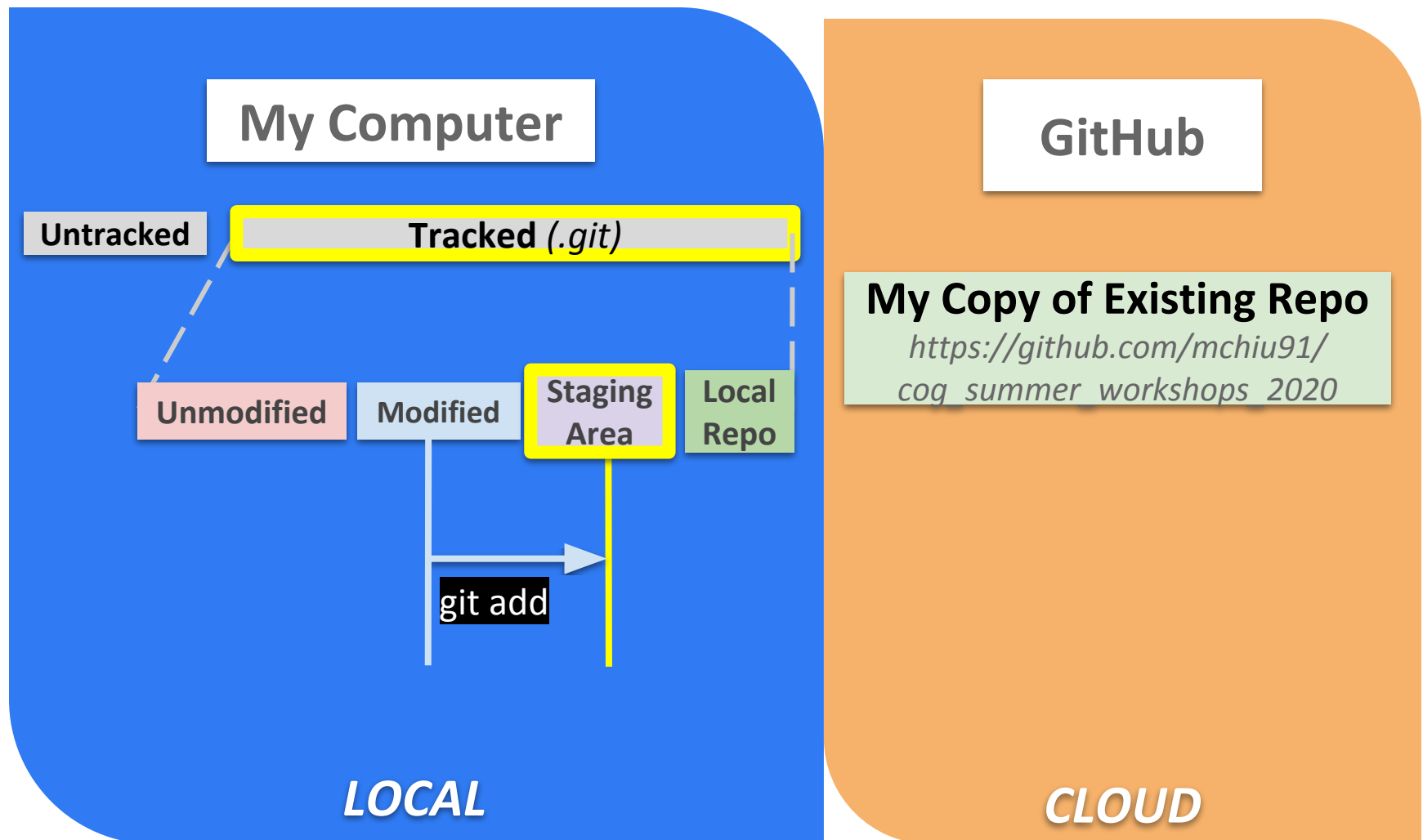
# Understanding Git File States



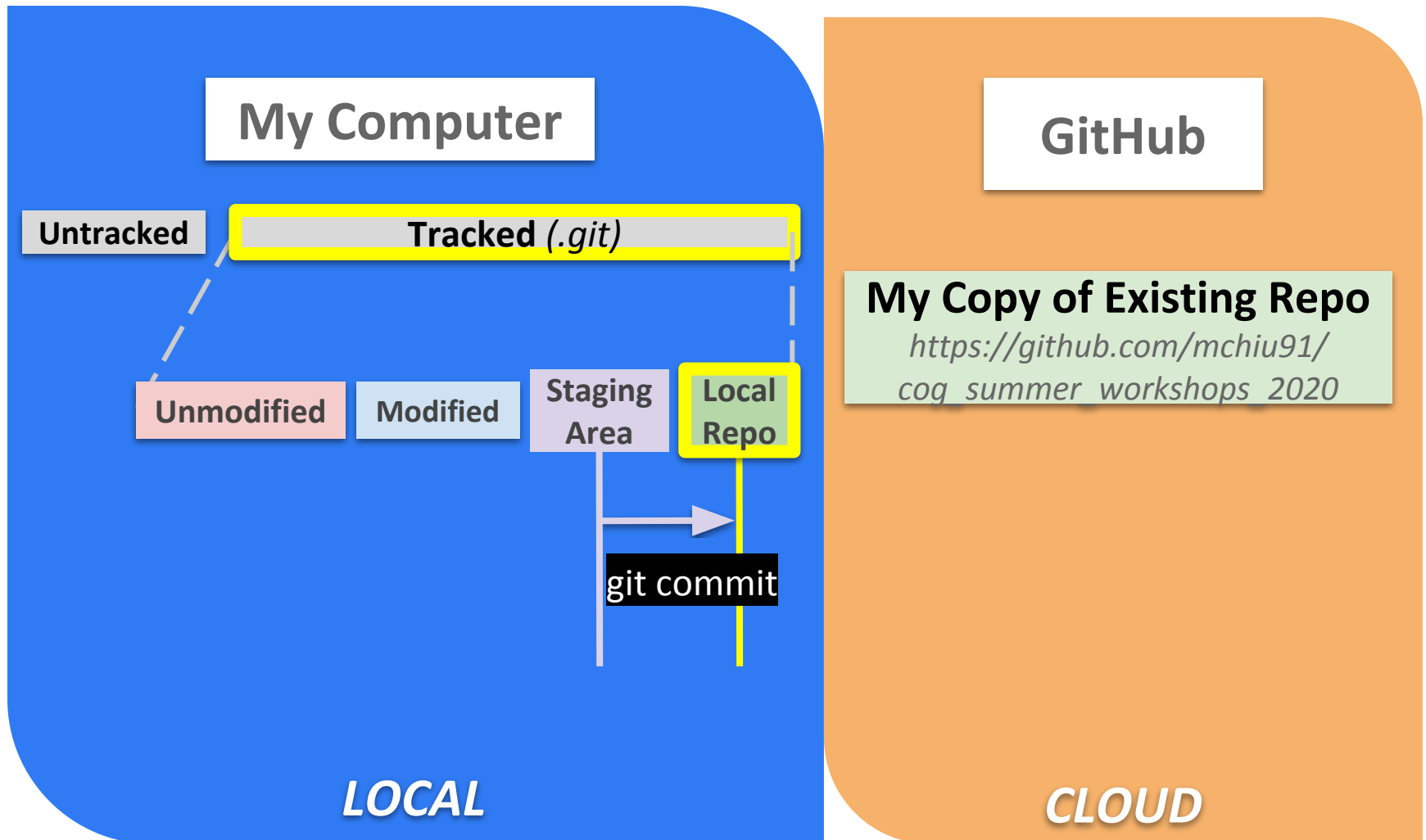
# Understanding Git File States



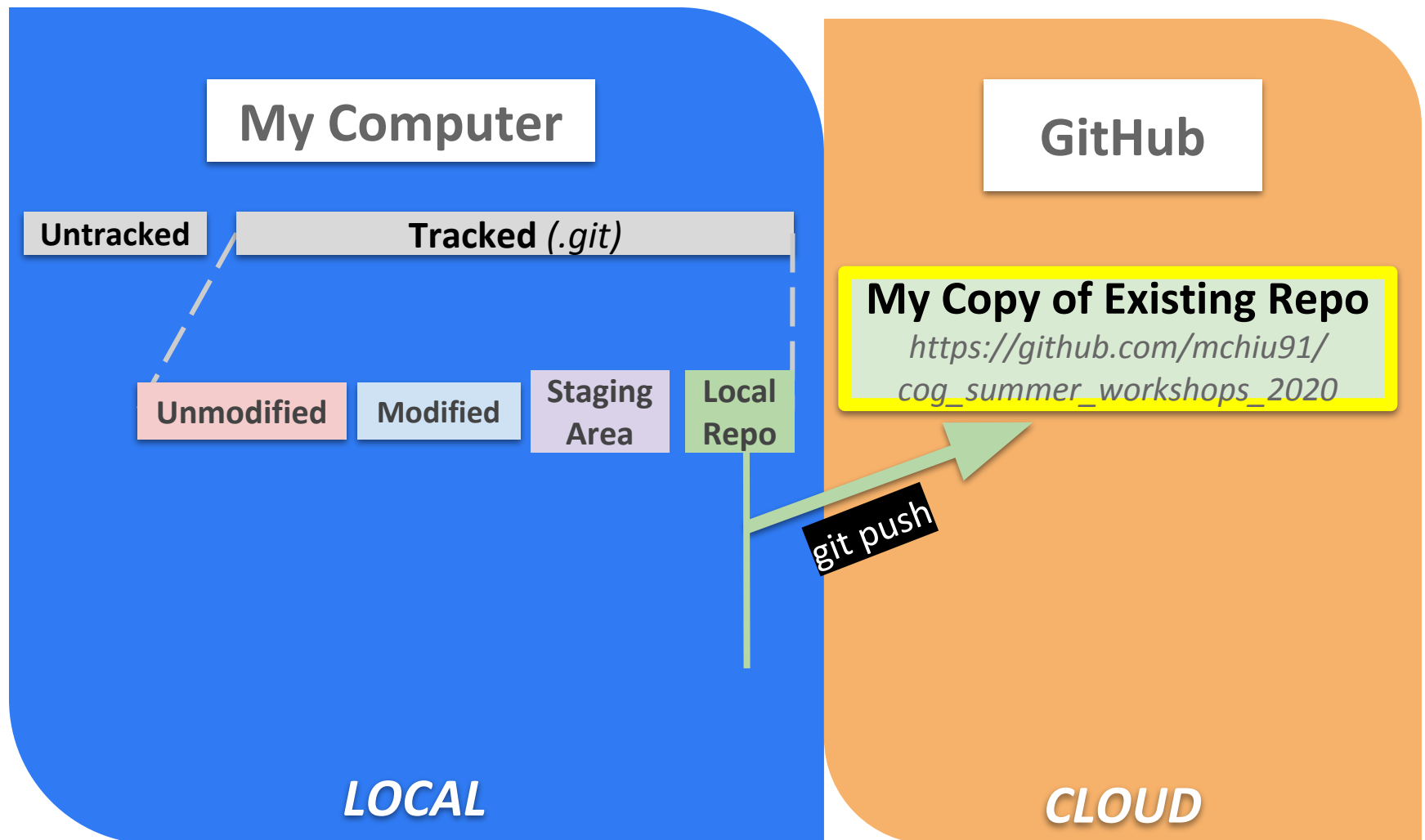
# Understanding Git File States



# Understanding Git File States



# Understanding Git File States







# Exercise 3: Add, Commit, Push


[https://youtu.be/s2WZ4TA\\_Obc](https://youtu.be/s2WZ4TA_Obc)




# Exercise 3: Add, Commit, Push

 **mchiu91 / cog\_summer\_workshops\_2020**

 Watch ▾ 0

 Star 0


 Fork 3


forked from [TU-Coding-Outreach-Group/cog\\_summer\\_workshops\\_2020](#)


[Code](#) [Pull requests 0](#) [Actions](#) [Projects 0](#) [Wiki](#) [Security 0](#) [Insights](#) [Settings](#)


No description, website, or topics provided.


[Manage topics](#)

 35 commits

 1 branch

 0 packages

 0 releases

 4 contributors

Branch: master ▾

New pull request


Create new file


Upload files


Find file

Clone or download ▾


This branch is 1 commit ahead of TU-Coding-Outreach-Group:master.

 Pull request

 Compare


 **Michelle Chiu** edited to include gritty info

Latest commit a1c7ea2 2 hours ago

 [git-github](#)


edited to include gritty info

2 hours ago

 [jupyter-notebook](#)

Add files via upload


9 days ago


 [README.md](#)


Update README.md


3 days ago

# Exercise 3: Add, Commit, Push

 **mchiu91 / cog\_summer\_workshops\_2020**

 Watch ▾ 0

 Star 0


 Fork 3


forked from [TU-Coding-Outreach-Group/cog\\_summer\\_workshops\\_2020](#)


[Code](#) [Pull requests 0](#) [Actions](#) [Projects 0](#) [Wiki](#) [Security 0](#) [Insights](#) [Settings](#)


No description, website, or topics provided.


[Manage topics](#)

 35 commits

 1 branch

 0 packages

 0 releases

 4 contributors

Branch: master ▾

New pull request


Create new file


Upload files

Find file


Clone or download ▾

This branch is 1 commit ahead of TU-Coding-Outreach-Group:master. [Pull request](#) [Compare](#)


 **Michelle Chiu** edited to include gritty info Latest commit a1c7ea2 2 hours ago

 [git-github](#)

edited to include gritty info 2 hours ago

 [jupyter-notebook](#)

Add files via upload 9 days ago

 [README.md](#)

Update README.md 3 days ago

# Useful Git commands

- **git status** shows the **status of a repository**
- **git add** puts files in the **staging area**
- **git commit** saves the staged content as a new commit in the local repository (always write a log message when committing changes)
- **git push** sends from local to GitHub
- **git log** shows the commit history
- **git diff** displays differences between commits
- **git checkout** recovers old versions of files

# Git Workflow

IN CASE OF FIRE



1. git commit



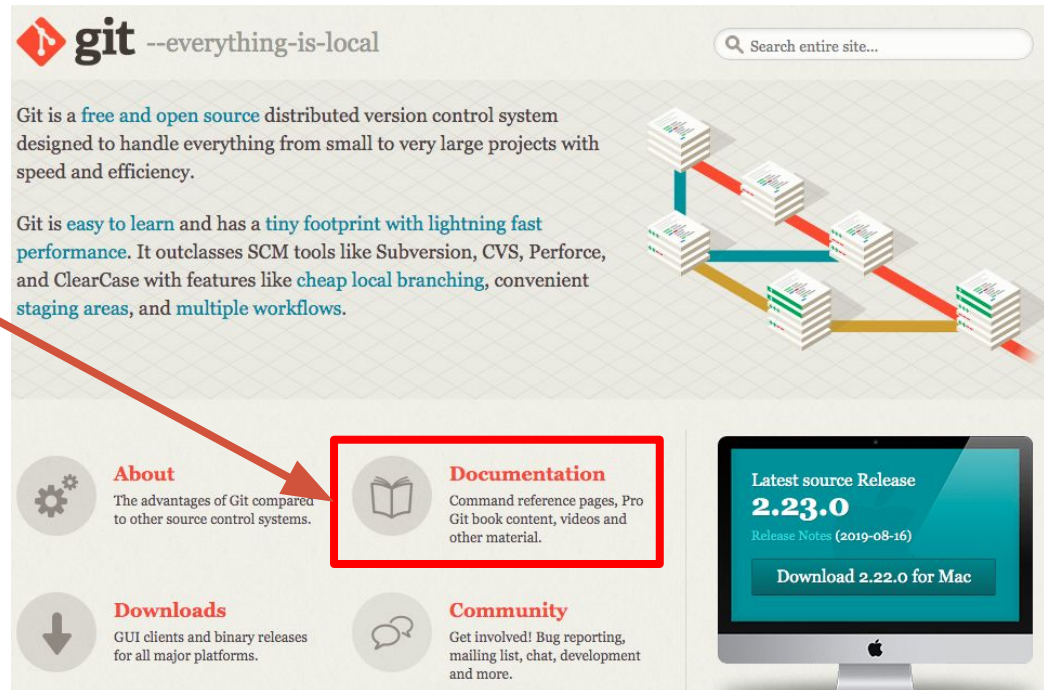
2. git push



3. git out!

# Resources (among many!)

- <https://git-scm.com>



# Resources (among many!)

- <https://git-scm.com/docs>  
(links to videos, static and interactive cheat sheets)

**GitHub**  
**GIT CHEAT SHEET**

Git is the open source distributed version control system that facilitates GitHub activities on your laptop or desktop. This cheat sheet summarizes commonly used Git command line instructions for quick reference.

**INSTALL GIT**  
GitHub provides desktop clients that include a graphical user interface for the most common repository actions and an automatically updating command line edition of Git for advanced scenarios.

**GitHub for Windows**  
<https://windows.github.com>

**GitHub for Mac**  
<https://mac.github.com>

Git distributions for Linux and POSIX systems are available on the official Git SCM web site.  
<http://git-scm.com>

**Git for All Platforms**  
<http://git-scm.com>

**CONFIGURE TOOLING**  
Configure user information for all local repositories

```
$ git config --global user.name "[name]"
```

**MAKE CHANGES**  
Review edits and craft a commit transaction

```
$ git status  
Lists all new or modified files to be committed  
$ git diff  
Shows file differences not yet staged  
$ git add [file]  
Snapshots the file in preparation for versioning  
$ git diff --staged  
Shows file differences between staging and the last file version  
$ git reset [file]  
Unstages the file, but preserve its contents  
$ git commit -m "[descriptive message]"  
Records file snapshots permanently in version history
```

**GIT CHEATSHEET**  
AN INTERACTION FROM NDP SOFTWARE

en fr es de

STASH      WORKSPACE      INDEX      LOCAL REPOSITORY

**status**  
diff  
add <file... or dir...>  
add -u  
rm <file(s)...>  
mv <file(s)...>  
checkout <files(s)... or dir>  
reset HEAD <file(s)...>

**reset --soft HEAD^**  
**diff --cached [<commit>]**  
**commit [-m 'msg']**  
**commit --amend**

**git status** Displays:  
• paths that have differences between the index file and the current HEAD commit,  
• paths that have differences between the workspace and the index file, and  
• paths in the workspace that are not tracked by git.

# Resources (among many!)

- <https://git-scm.com/docs>
- <https://try.github.io/>

## Resources to learn Git

### Learn by reading

#### Git Handbook

Git, GitHub, DVCS, oh my! Learn all the lingo and the basics of Git.

#### Cheat Sheets

Keep these handy! Reference sheets covering Git commands, features, SVN migrations, and bash. Available in a multiple languages.

### Learn by doing

#### Learn Git branching

Try Git commands right from your web browser. Featuring some of your soon-to-be favorites: branch, add, commit, merge, revert, cherry-pick, rebase!

#### Visualizing Git

Look under the hood! Explore how Git commands affect the structure of a repository within your web browser with a free explore mode, and some constructed scenarios.

#### Git-It

You've downloaded Git, now what? Download Git-It to your machine and you'll get a hands-on tutorial that teaches you to use Git right from your local environment, using commands on real repositories.



# Resources (among many!)

- <https://git-scm.com/docs>
- <https://try.github.io/>
- <http://dangitgit.com/>

**Dangit, I did something terribly wrong, please tell me git has a magic time machine!?!**

```
git reflog
# you will see a list of every thing you've
# done in git, across all branches!
# each one has an index HEAD@{index}
# find the one before you broke everything
git reset HEAD@{index}
# magic time machine
```

# Sources

- <https://doi.org/10.5281/zenodo.3369466> - slides by Stephanie DeCross (Harvard)
- <https://elizabeth-dupre.com/git-course/> - Git course by Elizabeth DuPre (Montreal Neurological Institute)
- <https://agripongit.vincenttunru.com/> ← Git tutorial with simple and clear visuals
- <https://codingbee.net/git/git-file-states> ← useful visuals for understanding file states