



# Collaborating Using GitHub ...



(GW Libraries) Informal Workshop  
February 12, 2018

[go.gwu.edu/gwlibgithubworkshop](https://go.gwu.edu/gwlibgithubworkshop) ("old" slides)

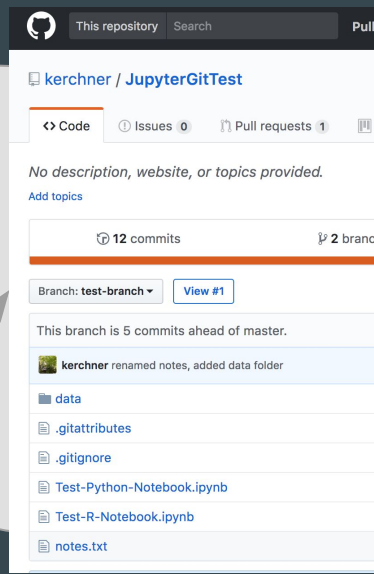
# Agenda

- Why GitHub?
- A tour of a repository
- Hands-on:
  - Create your own new repository
  - Collaborate on a repository
  - Fork an existing repository

# Why GitHub?

- Version Control
- Collaboration
- Open (or Not)
- Pricing - free for open, cheap for private, and free for students [education.github.com](https://education.github.com) (and [/pack](#))

Your repository on  
GitHub



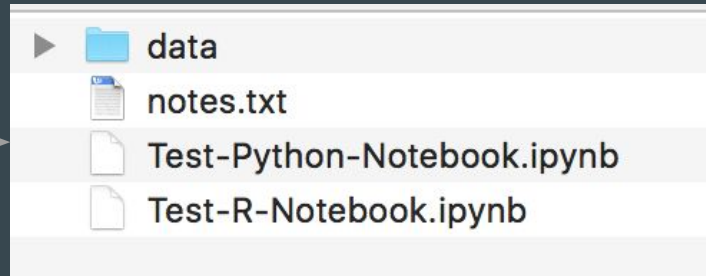
Apps you use that  
modify your files



RStudio

Microsoft Excel

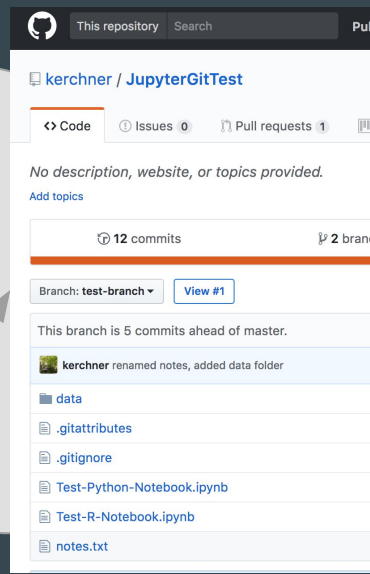
Files on your computer - your *local* git repository



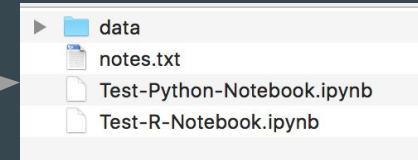
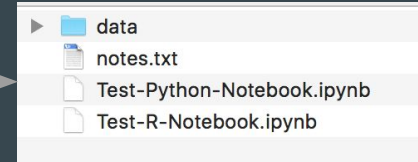
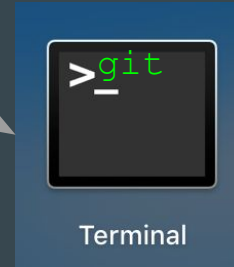
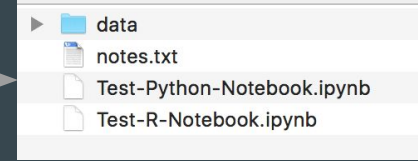
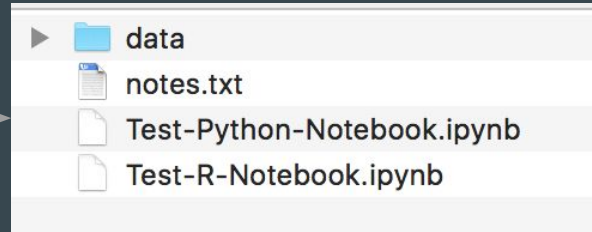
GitHub Desktop

*\* or command-line git, RStudio, Atom, etc.*

Your repository on  
GitHub



Collaborators' local copies of this repository



...or other apps with GitHub integration

# What makes a Git repository a Git repository?

- Open the folder containing your local git repository. Notice:
  - `.git` **directory** - contains all the metadata about the repository (that is, local copy)
  - `.gitignore` file (usually)
  - `.gitattributes` file (sometimes)

## Some Terminology

**Add/remove** - puts changes in a staging area

**Commit** - commits staged changes to the (local) repository

**Push** - pushes local repository changes to the remote repository

**Pull** - pulls remote repository changes to the local repository

# Tour of a Repository

- Features (just a few!)
  - README.md (showcase your work!)
  - [Markdown](#)
  - Code - history of commits on a branch, and on a file
  - Issues - tags, assignment, releases, comments
  - Watch, mentions, integrations
  - Pull requests
  - Milestones and Releases
  - Comments on issues, commits, etc.
  - Wiki
  - Branches
  - Insights (graphs)
  - Commits (and buttons next to each commit)
  - License



**Let's Try It!**

**Create your own GitHub repository**

# Create a repository

Two ways:

1. Create it on GitHub, then clone it.
2. In GitHub desktop, go to File→Add New Repository... then Publish it

**Collaborate on a repository**

# Collaborate on a Repository

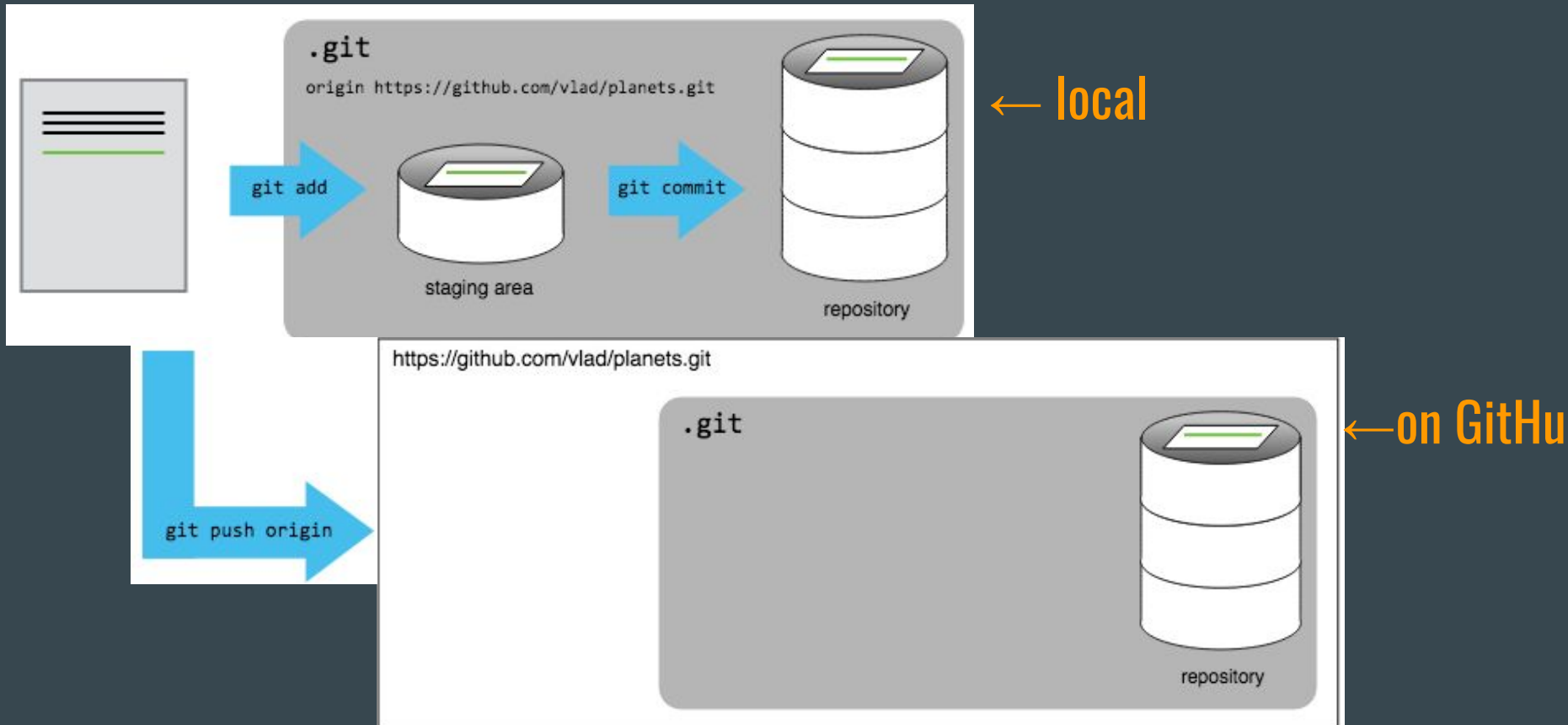
- One person: Create the repository, add collaborators
- Everyone else: Clone the repository
- Create some issues, assign tickets
- Work on issues in branches
  - Use commit messages like *'Adds title to plot. **Fixes #3**'*
- Push branches and create pull requests
- Review a pull request:
  - Check out someone else's branch and test it
  - Merge it if it looks good

# Fork a Repository

# Fork a repository

- Let's use <https://github.com/gwu-libraries/git-sandbox>
- First, fork on Github.com (press the Fork button)
  - Go to your repositories - notice that you now have a fork of git-sandbox
- Create a branch and modify some file(s)
- Create a pull request with this branch

# add ~ commit ~ push





# Advanced Topics

- Merges that require manual merging
- Resetting
- Re-basing
- Squashing
- Reverting

# Ways To Use Git

- Git command line
- GitHub Desktop App
- Github Mobile App
- Integrations w/Editors, IDEs, and other apps

# Git (Command Line)

```
git config --global user.email
git config --global user.password
git init      # creates a new repository based in this folder
git clone <URL for repo>
git status
git branch <branch-name>      # creates a branch
git checkout <branch-name>    # switches to a branch
git diff      /   git diff --staged
git add <file(s)>  /   git rm <file(s)>  /   git mv <file>
git commit -m 'Comment goes here'
git push origin <branch>      # When working with forks, you might
git pull origin <branch>      # use remotes other than "origin"
```

## Git (Command Line), continued

```
git log
```

```
git merge <branch to merge from>
```

```
git rebase <branch to rebase off of>
```

```
git reset --hard # Warning! You'll lose your changes!
```

```
git stash
```

```
git fetch # like git pull, but without merging
```

## Other Uses for Github Repositories (besides code)

- Sharing data
- "Document" collaboration (e.g. legislation)
- Course sites (e.g. [DSCN 6279](#); [ISTM 6212](#))
- Github Sites (e.g. <https://gwu-libraries.github.io/sfm-ui/> )
- Publishing and Citing your code (you can use Zenodo to mint a DOI for a Github repository)
- and more

## To Learn More...

- *Pro Git* book, free online: [git-scm.com](https://git-scm.com)
- Lynda.com: [\*Up and Running with Git and GitHub\*](#)
- github.com:
  - [help.github.com](https://help.github.com)
  - [guides.github.com](https://guides.github.com)
  - [try.github.io](https://try.github.io) (codeschool)
  - [services.github.com/on-demand/](https://services.github.com/on-demand/)
  - [resources.github.com/webcasts/](https://resources.github.com/webcasts/)

## A few takeaways from today

- The GitHub Desktop app seems useful, but we had at least one case of it freezing up. It also is useful mainly for "basic" things. Ultimately, it's probably good to learn and use git shell commands. (Especially good on your resume!)
- For team projects, one person can create the repository on their git account and add others as collaborators. (Or, you can create an "organization" account.)

## More takeaways

- Insights → Network is a good way to visualize what's going on with the branches.
- When working together (or even alone), use branches.  
Merge a branch using a pull request.
- When you're working on a branch for an extended period, consider merging (or rebasing) from the master branch, so your branch doesn't diverge too far from the code on master.



## Even more takeaways

- This workshop may have been a bit informal, there was a lot to cover, and the room was hot. You are likely to have questions when you try this on your own. If you get stuck, contact me and I will try to help.

# Thanks!

- Dan Kerchner     [kerchner@gwu.edu](mailto:kerchner@gwu.edu)

Coding Consultations: [go.gwu.edu/coding](https://go.gwu.edu/coding)