Introduction to GitHub

Julia Stewart Lowndes Project Scientist, Ocean Health Index
lowndes@nceas.ucsb.edu

What is GitHub?

GitHub is an open-source development platform that enables easy collaboration and versioning, which means that all saved versions are archived and attributed to each user. It is possible to revert back to any previous version, which is incredibly useful to not only to document what work has been done, but how it differs from work done in the past, and who is responsible for the changes.

Similar to Dropbox, you have certain folders on your local computer that will be 'watched', with any changes able to be synched online. However, with GitHub, you have more control about what is synched, and how often. You can store, share, track changes and collaboratively edit many filetypes (including this presentation!), using any program to edit. There are many other great features, including a to-do list you can share with collaborators (called Issues).

Why use GitHub?

There are so many reasons to use GitHub. Personal organization, switching between your home and work computers, backing up, version control. Also for collaboration, sharing, learning, contributing. . .

Nicely explained by Hadley Wickham: Git and GitHub Tutorial

And also by Karl Broman: GitHub Tutorial

And also by Ben Best: ds-git Tutorial

Workshop Outline

1. GitHub Structure
2. GitHub Workflow and Vocabulary
3. Best Practices
4. Cloning and Synching Options
5. Workflow and Practice Using RStudio
6. Resources

GitHub Structure

GitHub stores files in **repositories**, owned by an **organization**.

Repositories ('repos') are essentially folders containing files pertaining to a specific project. Repositories are version controlled so that any modifications to files, additions or deletions, are tracked and attributed to contributors with the correct permissions.

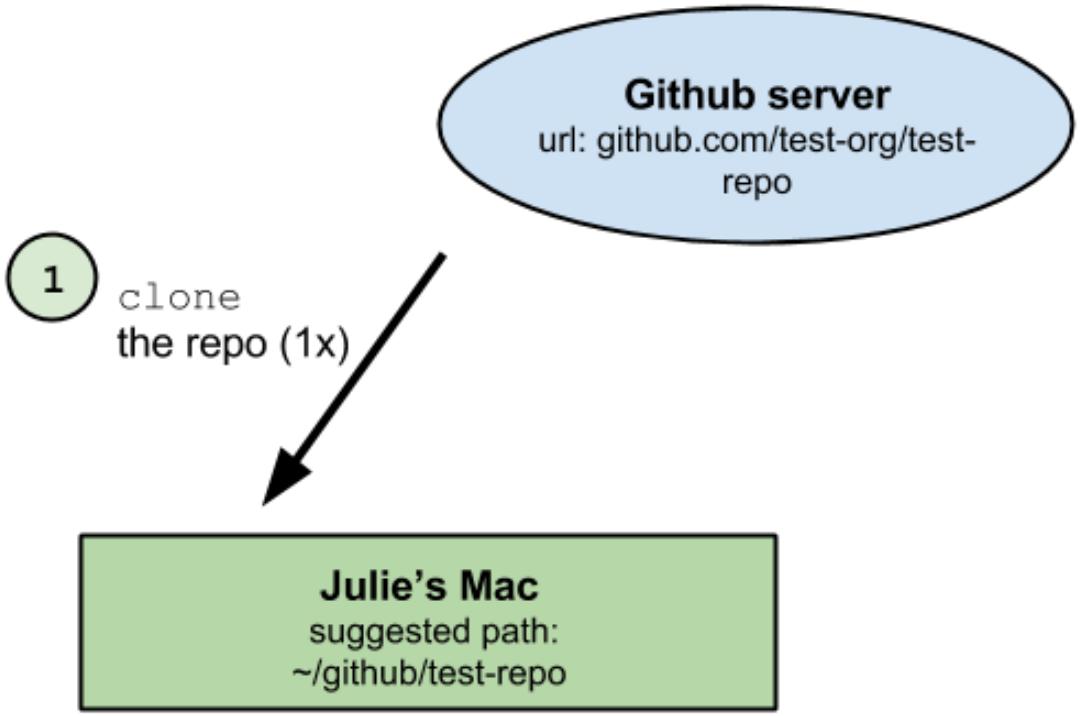
Same structure across all orgs/repos: familiar, easy to navigate.

Let's explore a bit:

- **organizations**: twitter, netflix, rstudio, nceas, ohi-science
- **repositories**: github-intro, dplyr, ggplot2

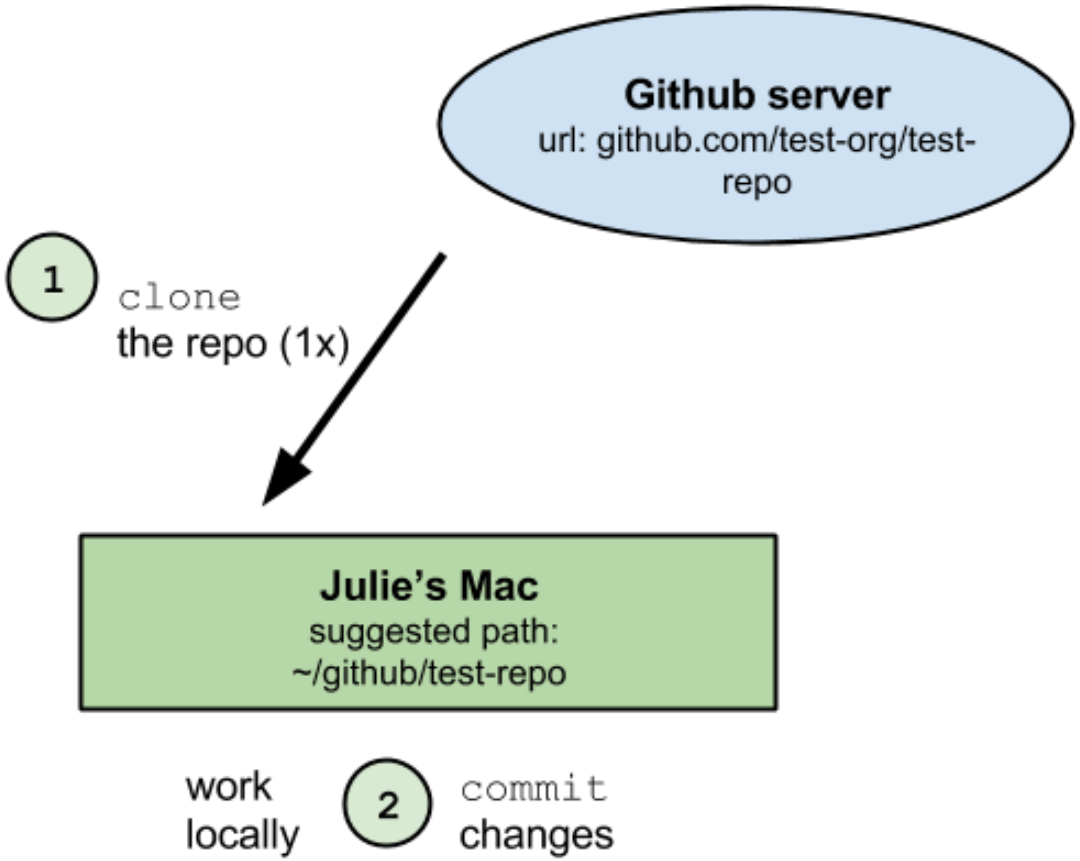
GitHub Workflow and Vocabulary

- **clone**: download to your computer from online version with synching capabilities enabled



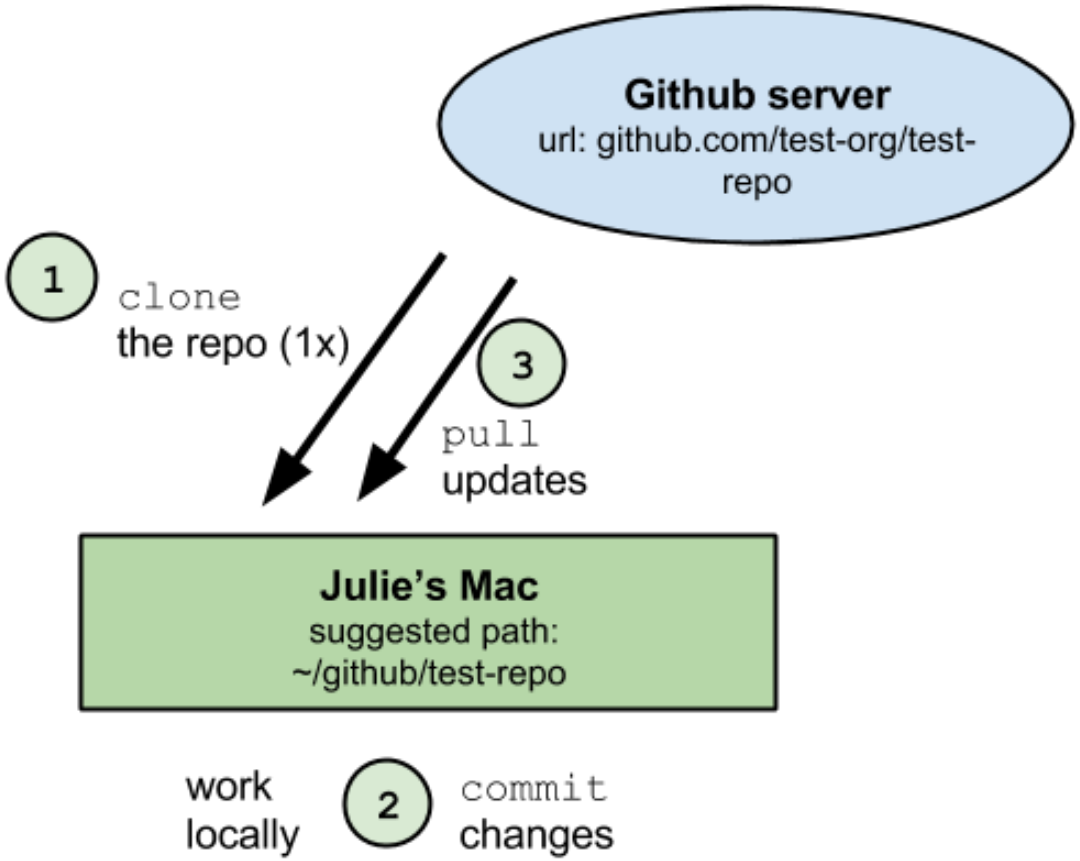
GitHub Workflow and Vocabulary

- **commit**: message associated with your changes (best practices)



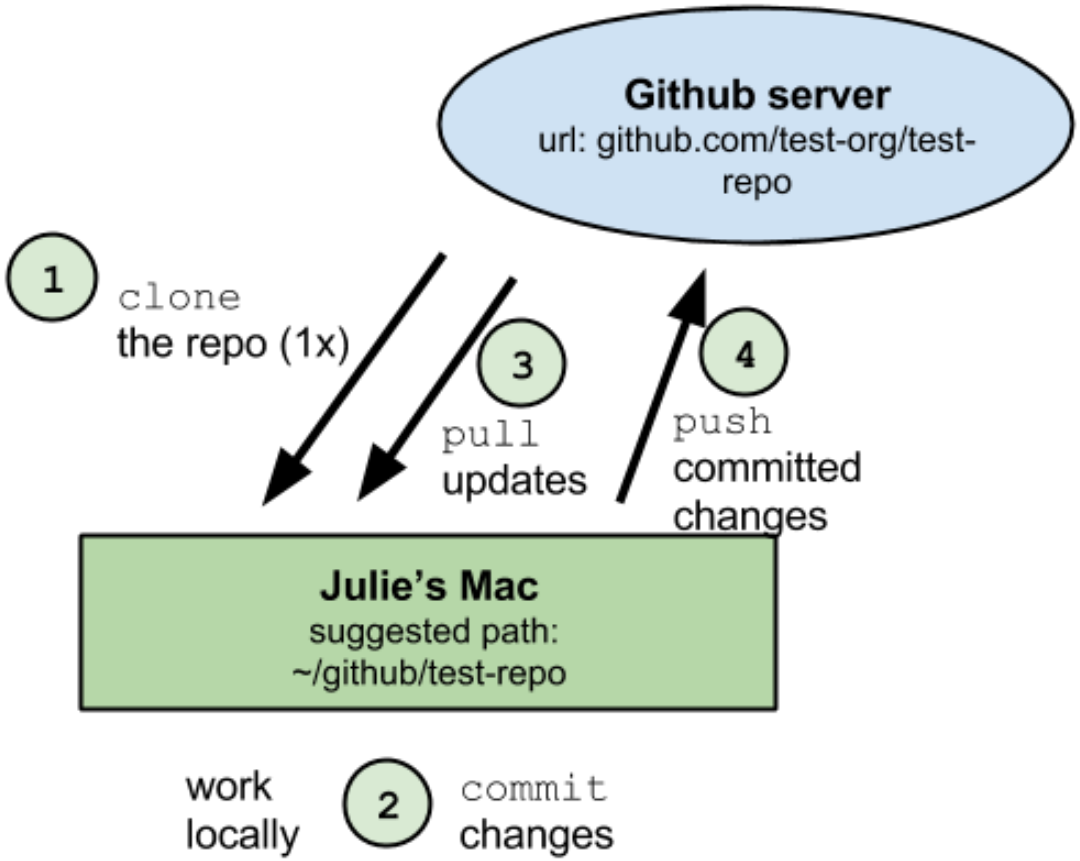
GitHub Workflow and Vocabulary

- **pull**: sync a repo on your computer with the online version. Do this frequently.



GitHub Workflow and Vocabulary

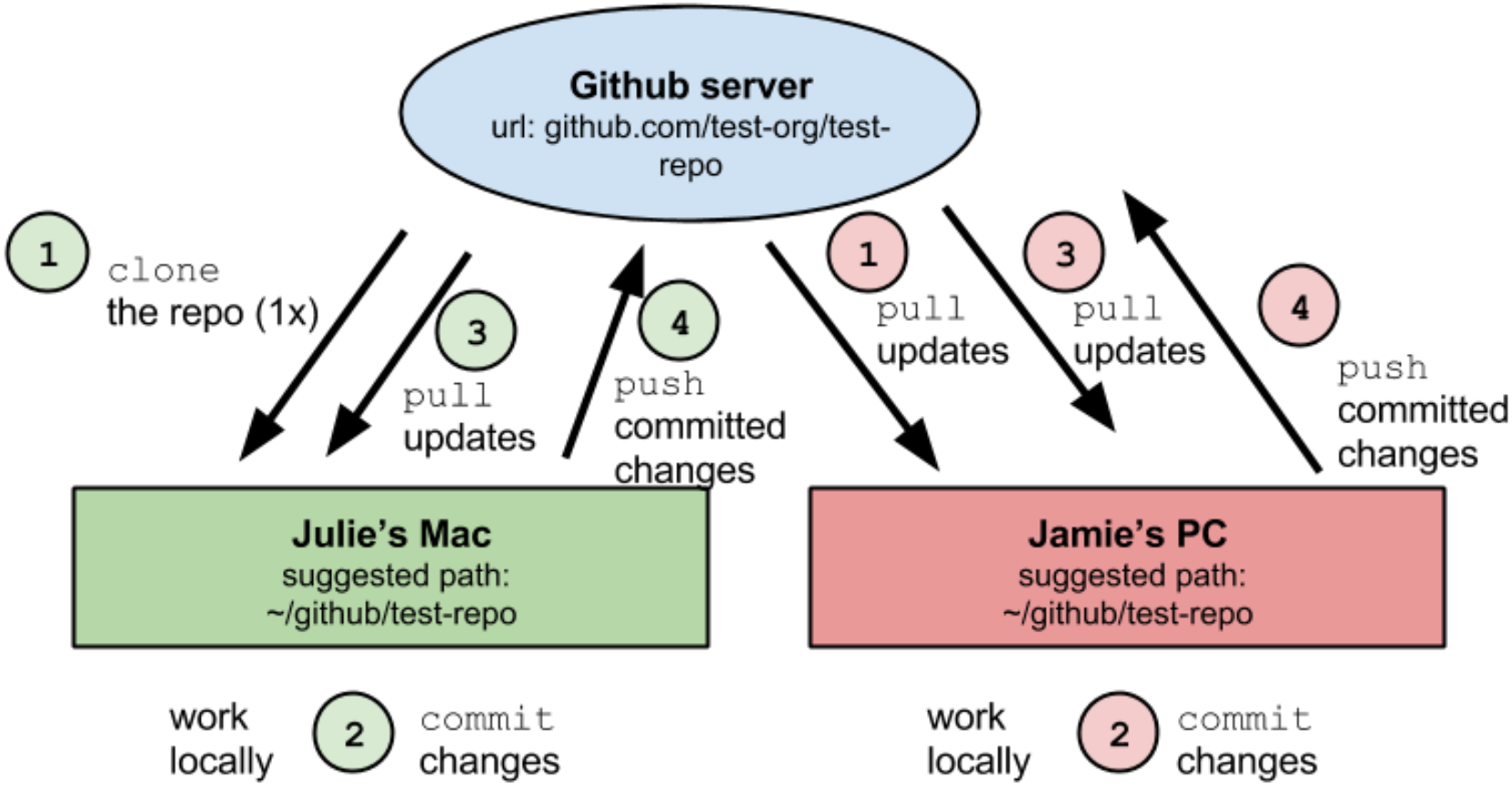
- **push**: sync the online repo with your version, only possible after committing



GitHub Workflow and Vocabulary

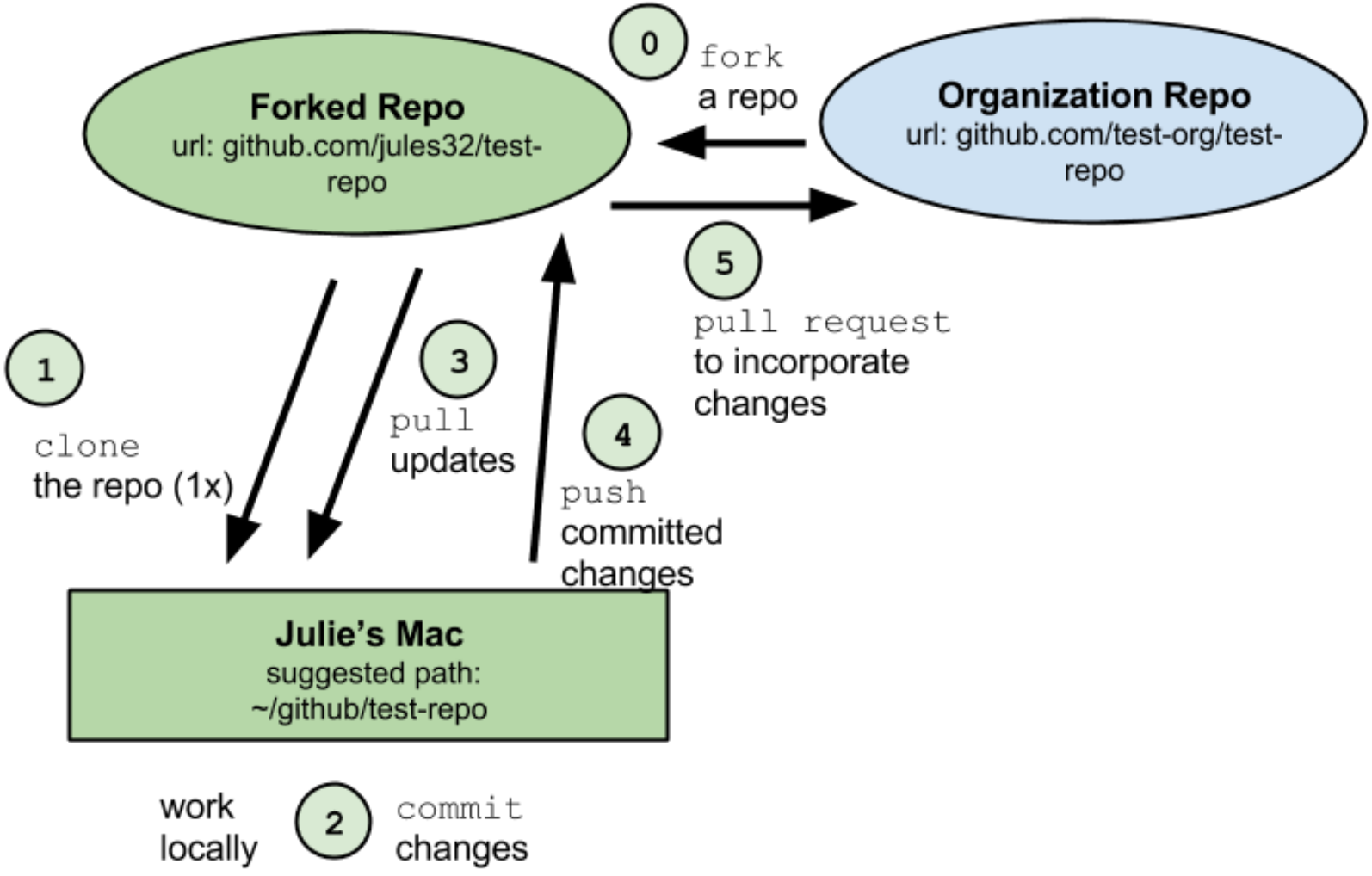
sync ~ pull + commit + pull + push

All collaborators work independently but sync regularly



GitHub Workflow: fork and pull

fork + pull + commit + push + pull request



Best Practices

Pull often!

Commit frequently

Be mindful of filepaths

- We work from a in a folder in our home directory called ***'github'*** (all lowercase!), so that everyone can access the repo with the filepath beginning in `~/github`:
- **Windows**: `Users\[User]\Documents\github\`
- **Mac**: `Users/[User]/github/`
- *Please make a `github` folder in your home directory.*

Cloning and Synching Options

You will clone a repo to your computer and work locally.

When you work on your computer, any edits you make to any files in your repo, using any program, will be tracked by GitHub. You can then commit and sync your changes back to GitHub. There are many options you can use to first clone and then sync your edits on a repo with the online version:

- **GitHub App** for Mac and for Windows
- **RStudio**
- **shell (Terminal on Mac)**

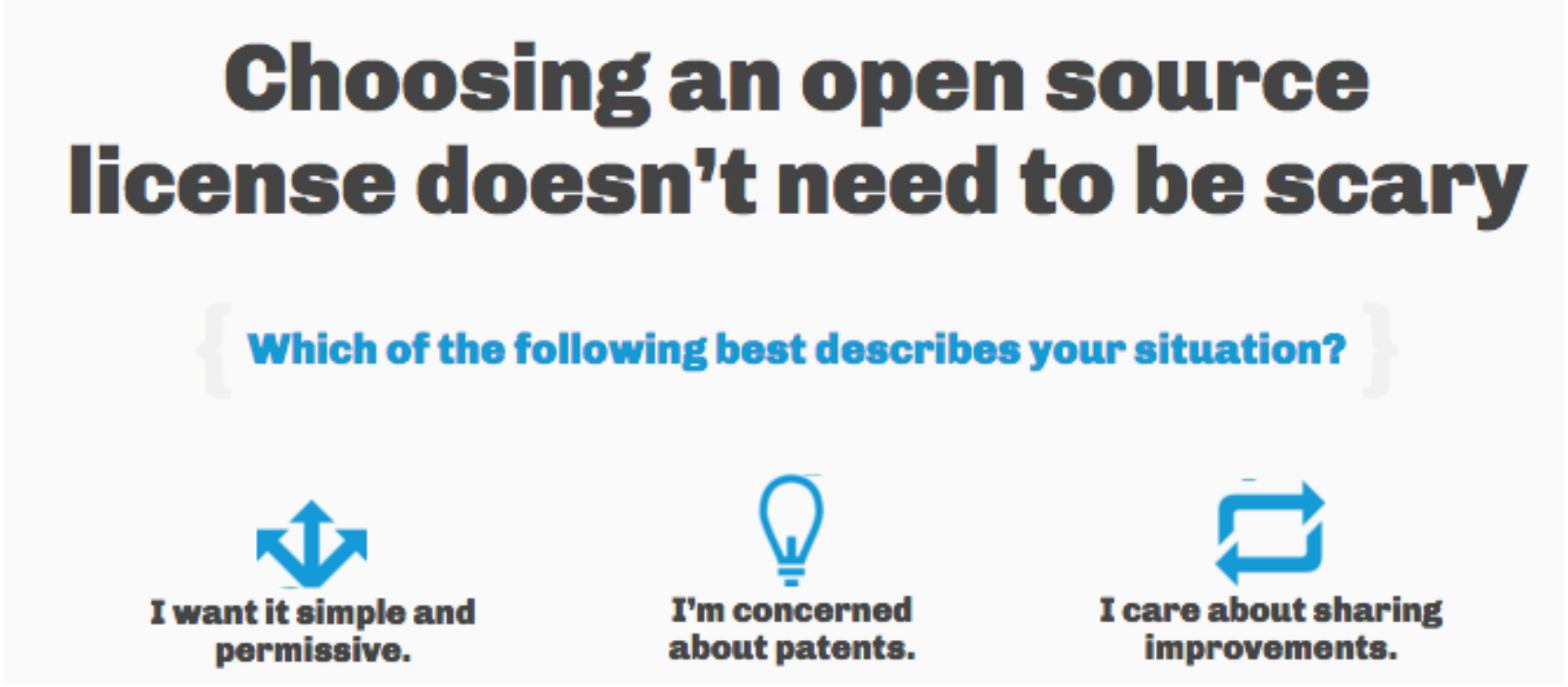
We will just use RStudio today.

Workflow and Practice Using RStudio

1. **fork a repo** to your user account
2. **clone the repo** to your computer
3. **edit a file**, ex github-intro/test_script_ohi-uswest.Rmd
4. inspect differences
5. **commit** changes
6. **pull**
7. **push**
8. repeat!

GitHub and science (and beyond)

- GitHub gets its science on
- Making your code citable
  - Choosing an open source license



- Ways to use GitHub that aren't coding

Resources

Learn more about GitHub:

- **Git and GitHub** by Hadley Wickham
- **Good Resources for Learning Git and GitHub** by GitHub
- **Learn Git Branching** by Peter Cottle
- **Git/GitHub Guide** by Karl Broman
- **Git & GitHub** by Ben Best
- **Hello World GitHub Guide**, a 10-minute tutorial by GitHub

Just Google 'GitHub Tutorial...'