```
library(dplyr)

rladies_global %>%
  filter(city == 'Charlottesville')
```

Git & Github for the UseR

November 5, 2019

Workshop materials:
http://bit.ly/cville_github

Welcome!

- WiFi: cityguest (no password)
- Restrooms located down the hall
- Workshop materials: http://bit.ly/cville_github

Sponsors + Friends (for now)

RStudio

consortium

HEMOSHEAR THERAPEUTICS

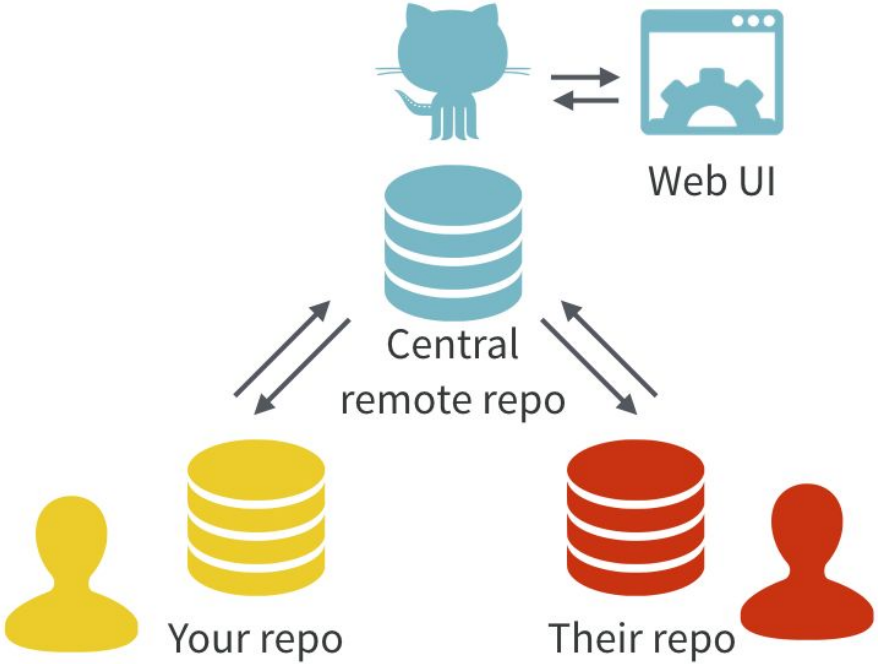
VividCortex

Want to support another way? Give a talk!

Agenda for today:

- Intro to Git & Github
- Demo & code along
- Questions & Resources

Excuse me, do you have a moment to talk about version control?



https://doi.org/10.7287/peerj.preprints.3159v2

Your turn: tour a repository

Spend a couple minutes exploring these repos:

https://github.com/brooke-watson/BRRR

https://github.com/jayqi/spongebob/

What's a README? What are commits? Issues?

If you don't have a Github account of your own, make one now.

1.
What is  git ?

What is Git?

- **Git** is a free and open sourced distributed version control system.
- It manages the evolution of set of files, or **repositories**, between local and online versions
- It allows other people to see your stuff, sync up with you, and perhaps even make changes.

Your turn: *Git* started

Do I already have Git?

```
which git
```

What version of Git do I have?

```
git --version
```

Eek! I don't have Git!
Fear not- time to install!

https://www.atlassian.com/git/tutorials/install-git

Your turn: introduce yourself

Hi Git!

In Terminal:

```
git config --global user.name 'Samantha Toet'
git config --global user.email 'samanthactoet@gmail.com'
git config --global --list
```

Use the email address your github account uses.

2.
What is  GitHub?

What is Github?

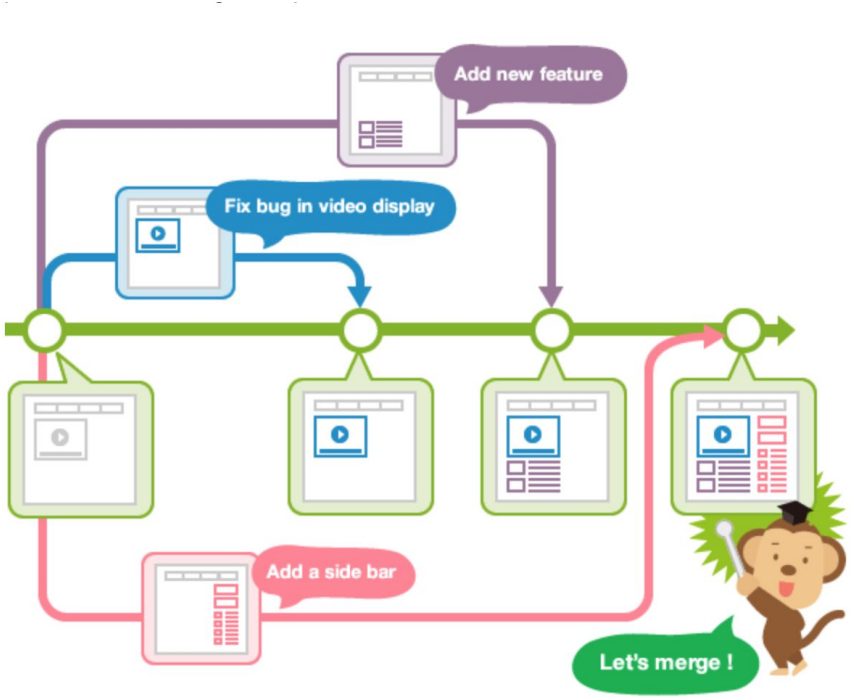
- Github is a web-based hosting service for version control – it's where the online backup of your local project lives.

- Github allows a project to have multiple, independent **branches** of development

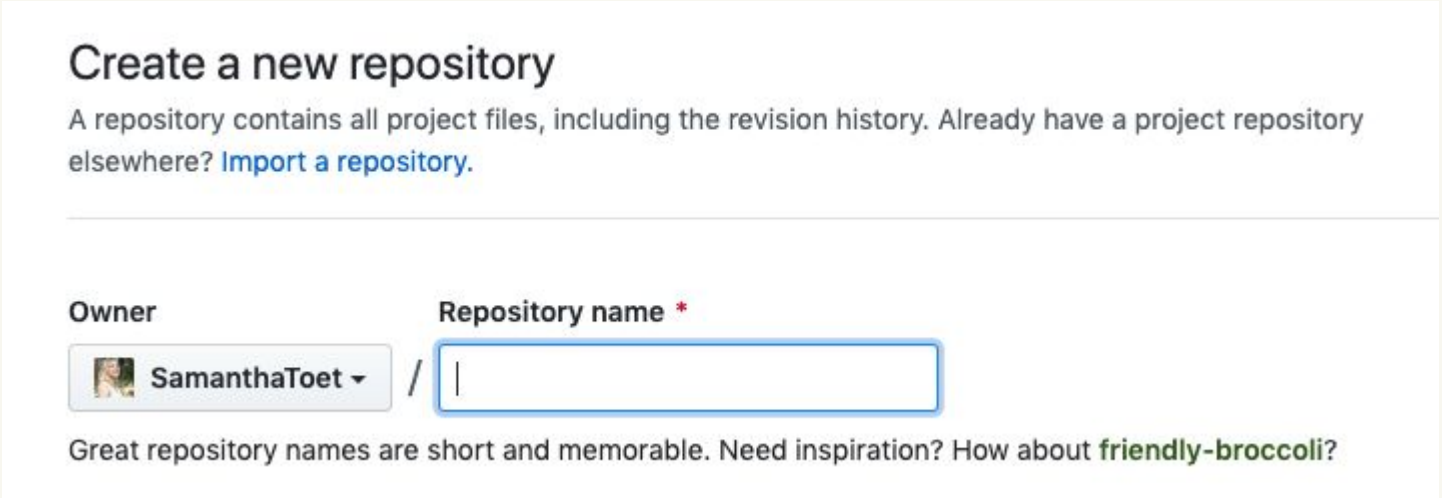
- For collaboration, there are two other important features:
    - **Issues**: list of bugs, feature requests, to do's, etc.
    - **Pull requests**: a formal proposal that says "Here are some changes I would like to make." (It might be linked to a specific issue)

Terminology

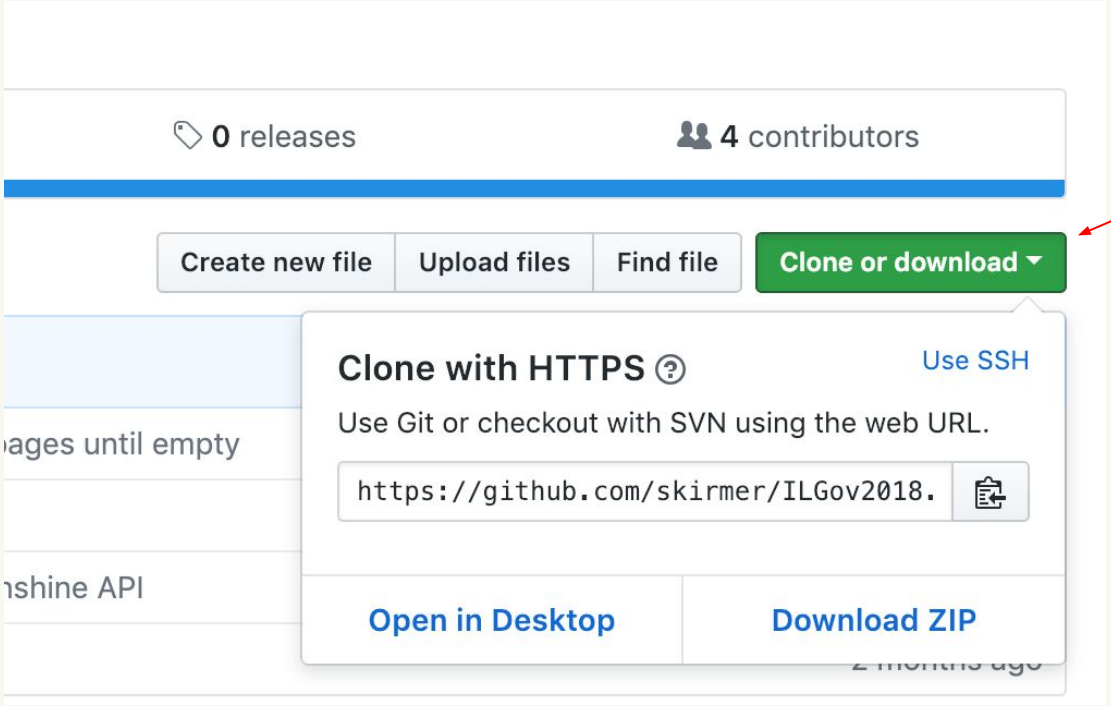
- **Local** (on your computer) vs. **Remote** (on the internet)
- **Repository**: a place where the history of your work is stored
- **Branch**: a way to maintain different versions of the same code base. When you create a branch in your project, you're creating an environment where you can try out new ideas and if they work, you can **merge** them back to the master



Your turn: create a repo



Your turn: clone the repo



This link tells your computer where to find the **remote** location.

Your turn: bring it home

Now, in your local terminal, type:


```
git clone
https://github.com/YOUR-USERNAME/YOUR-REPOSITORY.git
```

(*Hint: use your Desktop as your working directory for easier navigation and folder clean up)

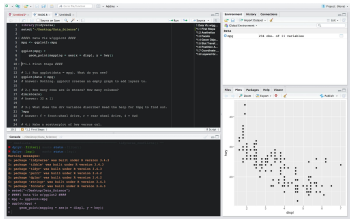
This creates a partner folder in your local computer that is paired with the remote repository.

Git knows these two folders are paired, and can do many useful tasks to compare them and keep them consistent.

How does it work?

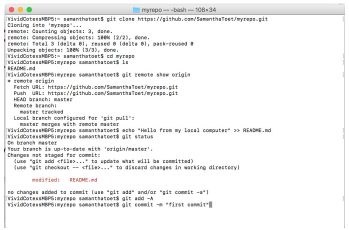
RStudio

- Used locally
- Where the coding magic happens
- Files ***saved*** in folders



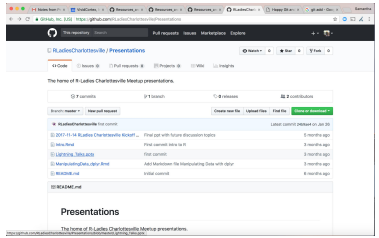
Git

- Used at the command line
- Repository = folder
- ***Committing*** saved local files to staging (i.e. saving as multi-file snapshot)



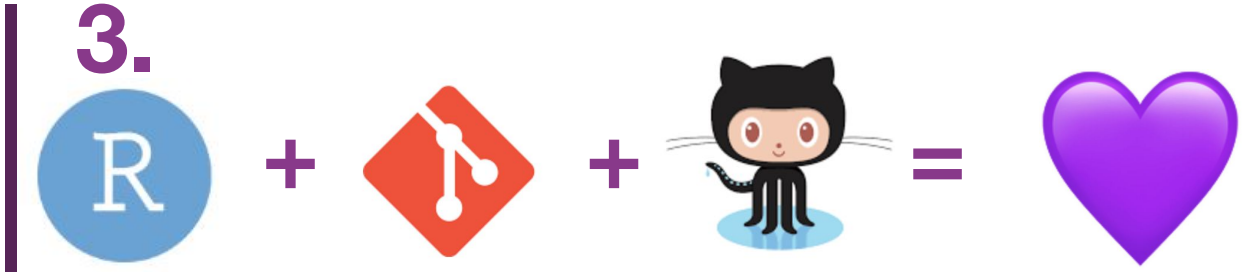
Github

- Used online
- ***Push*** your local ***commits*** to Github periodically
- Share your work and collaborate with others



Quick Git Workflow

- What's up with my repo?
  `git status`
- I want to queue a file up to be sent to "remote"
  `git add file_name.extension`
- I'm ready to package the file with a note to be sent to "remote":
  `git commit -m "message content!"`
- Send these added files to "remote"!
  `git push`


3.
R
+
+
=

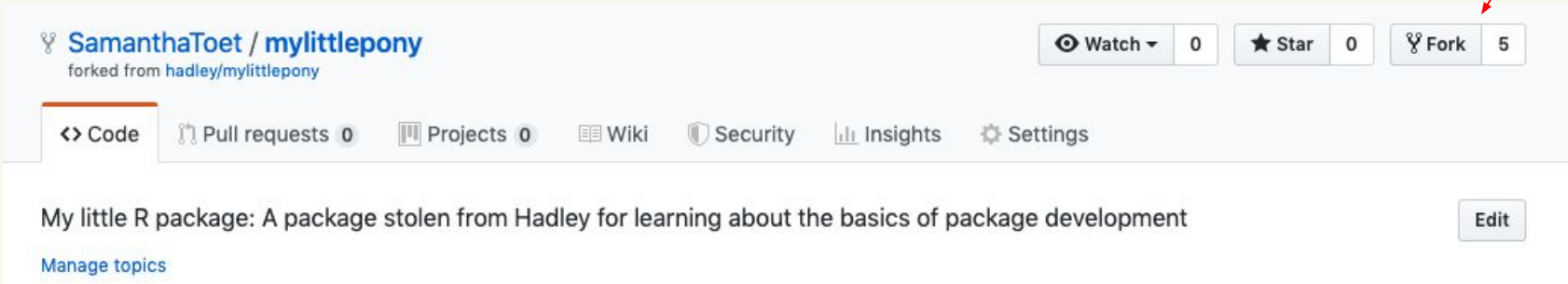
Github Repository = RStudio Project = (your own) Package

- When working with multiple projects and repos, it's generally easier to associate each project with a different repository.

- This also helps streamline the package development process.

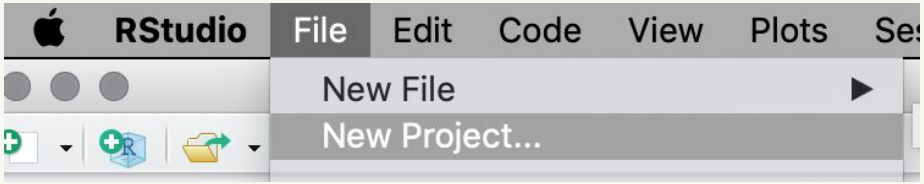
Your turn: My Little R Package

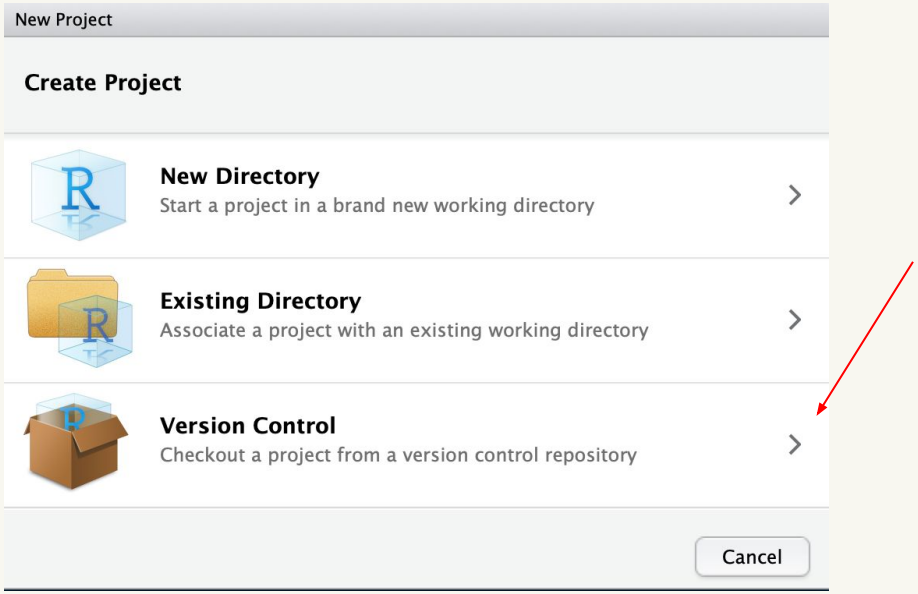
1. Go to github.com/SamanthaToet/mylittlepony
2. Fork the package to your repo



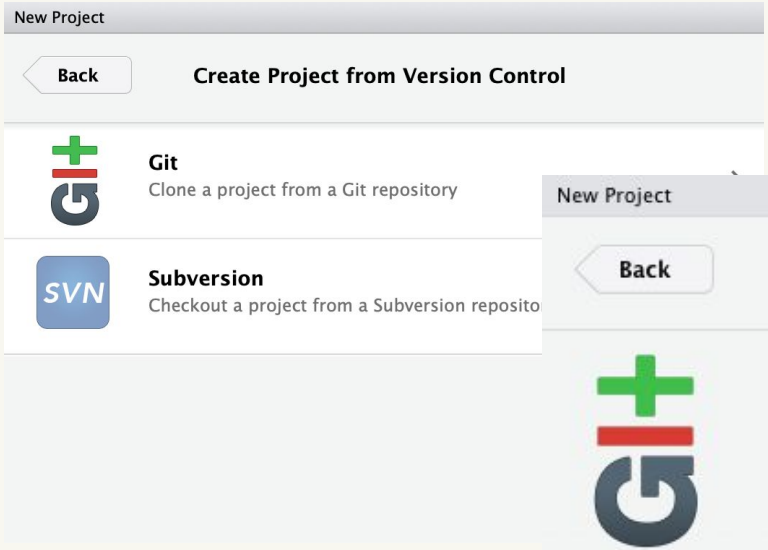
3. Go to your forked version of the repository and clone it

Your turn: create a project from version control

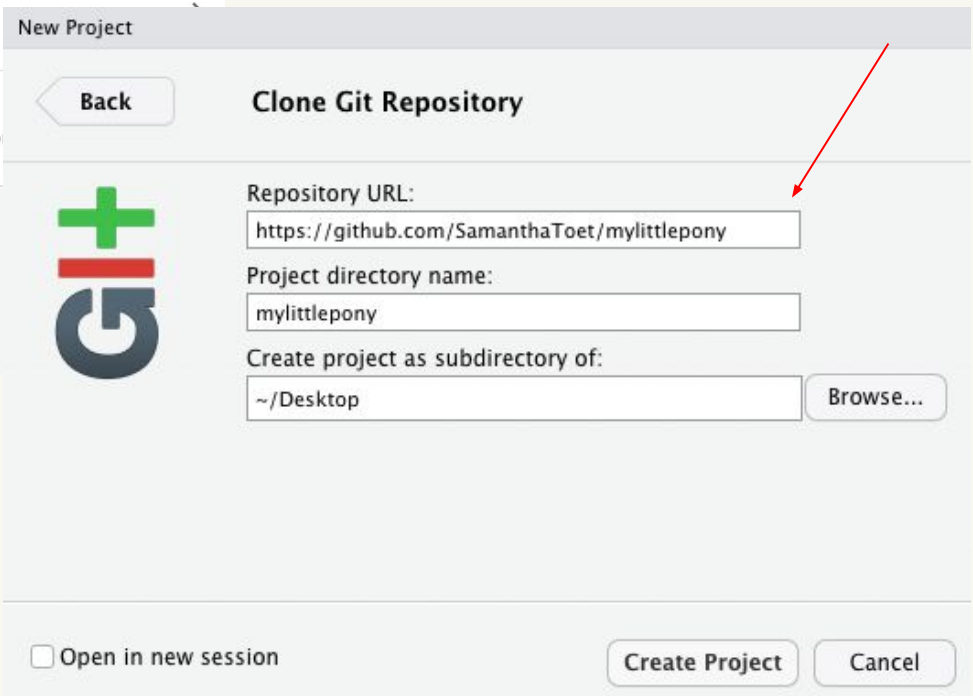




Your turn: create a project from version control



Paste the url of the new forked repo here. It should look like: https://github.com/YOURNAME/mylittlepony



Your turn: explore this new project

Let’s look around and see how this new project works.

What’s the Git panel? History? New branches?

Your turn: collaborate on a project

1. Open mylittlepony.Rproj
2. Open rpony.R (What is this file? What does it do?)
3. Load all the functions by pressing CMD + SHIFT + L, then run `rpony(10)`

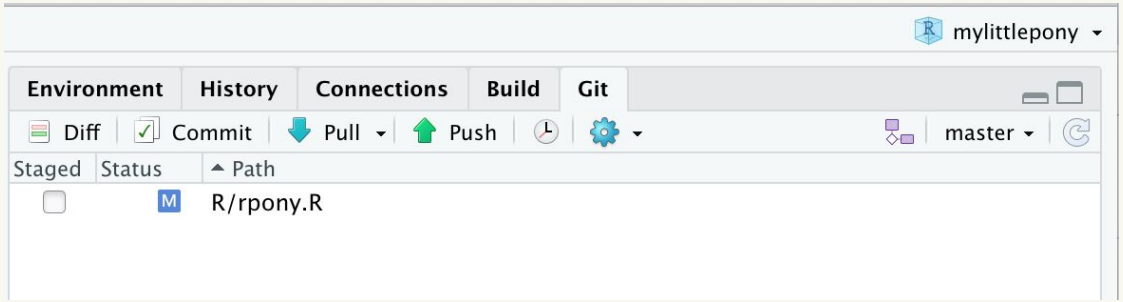
What does it do?

Uh oh! I have forgotten to include *Fluttershy* in the list of ponies. Add her, reload the code, and verify that your change worked.

Your turn: *Your* Little R Package

1. Once you added *Fluttershy* in the last section, push your changes to Github (*Hint: remember the mailing analogy, **add** the gift to a box, **commit** it with a gift message, and **push** it to the post office)

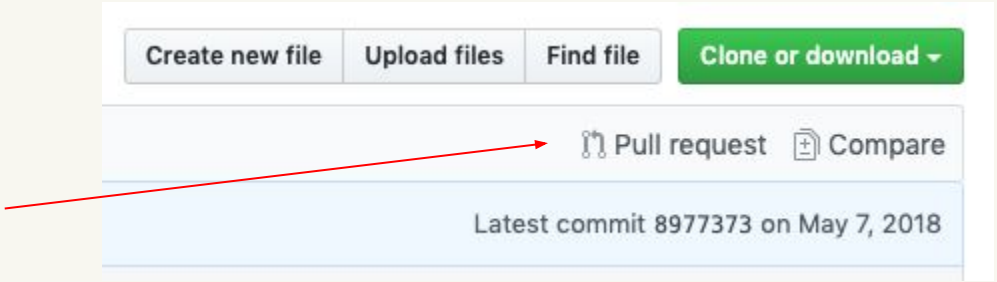
   (**Hint: you can do this manually in the terminal, or you can do this in the Git pane of the RStudio console)



2. Go to your Github and look at the changes to your repository.

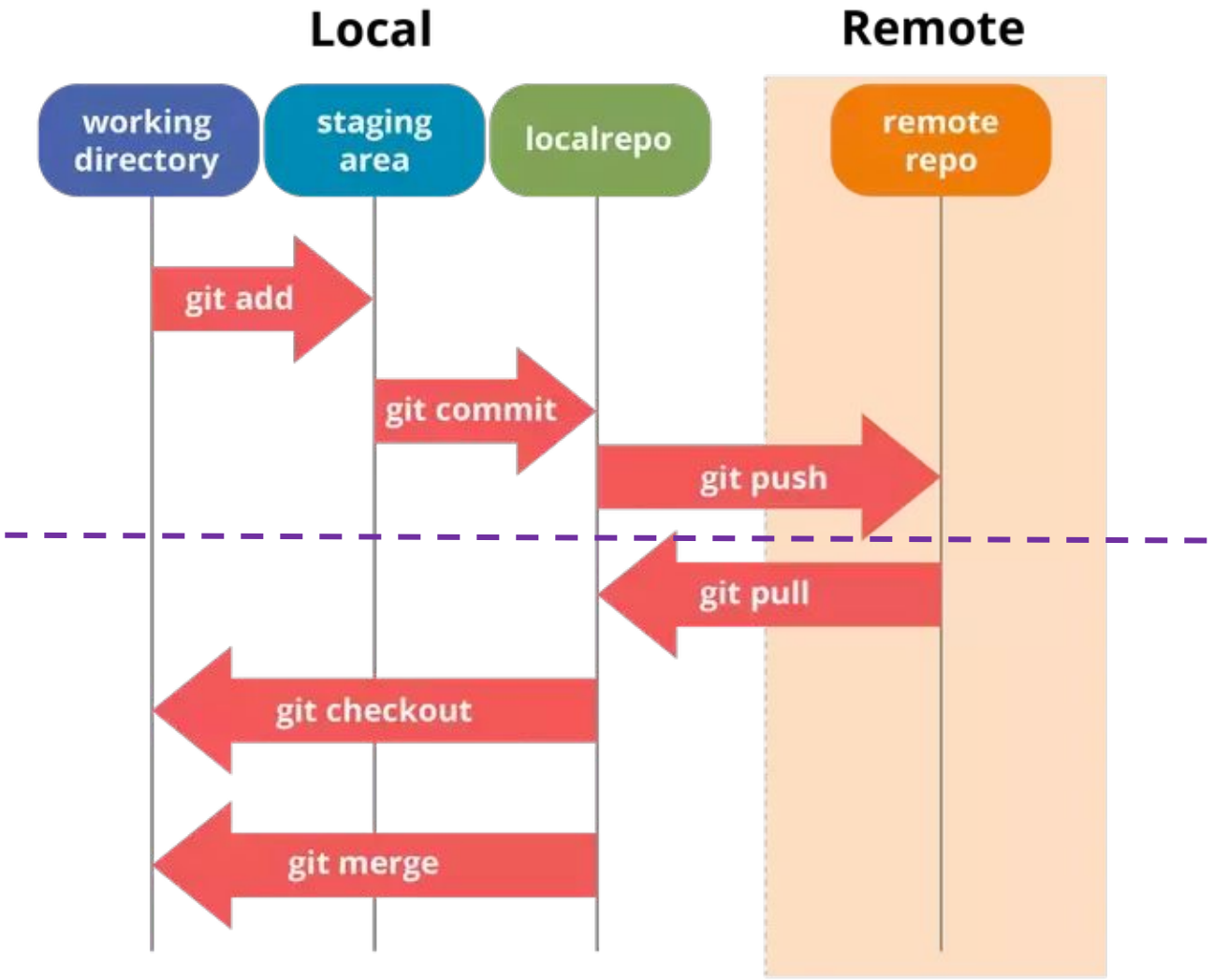
Your turn: make a pull request

Your repository is now different from mine. To share the changes you made with me you'll need to submit a pull request. Once I approve the pull request I'll merge these changes into my original repository.



Congrats!

You have successfully contributed to an open source project!

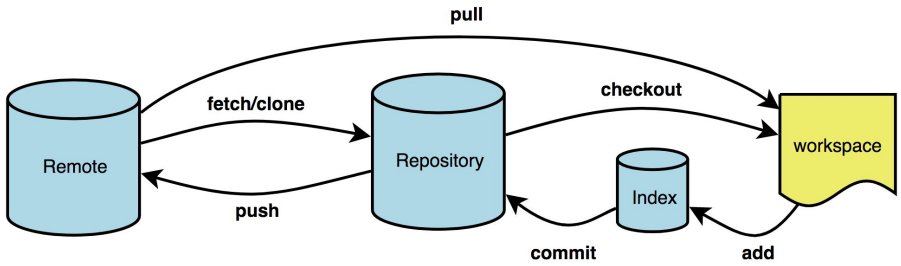

Local
Remote
working directory
staging area
localrepo
remote repo
git add
git commit
git push
git pull
git checkout
git merge

3. Troubleshooting & Resources

Collaborating

In a collaborative repo, the origin/master might be ahead of yours so you can't just push your content to it. So what do you do?

This is where things can get messy.



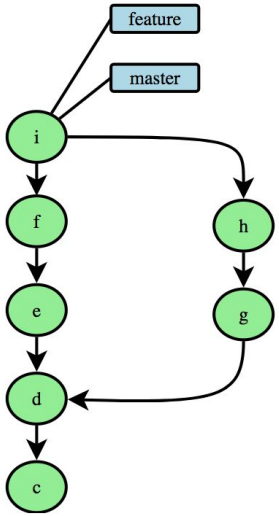
Tips on collaboration

Make sure you understand how you're getting updates:

- **Forking** (Github, not Git) – make a personal copy of another repo that lives in your Github. You can make changes to the project without affecting the original, and keep it up to date by **pulling** changes from the original.

- **Fetching** - get the latest changes from an online repo without merging them in. Once these changes are fetched you can compare them to your local branches.

- **Pulling** (`git pull`) - fetch the latest from the remote and merge them into you changes. Note: this might result in a merge conflict that has to be resolved, as it will commit your changes on top of what others have done.



More tips on collaboration

Make sure you're comfortable moving around repositories:

- Use `git diff` to check the difference between the remote and the local repo
- For retrieving a repository that you don't have, use `git clone`
- For switching between branches in a repo you already have, use `git checkout`
- Don't push to master, even on your own repositories. Use a separate branch. Why do you think this is a best practice?

Troubleshooting – there's gonna be a lot of it

- Explore history:
  - `git status` checks the status of your current commits
  - `git log` shows all latest commit logs
  - `git log --oneline --graph` shows a graphical representation including branching
  - `git log --follow file.ext` shows version history of a file
- Try some resets:
  - `git reset <commit id>` resets to specific commit, changes are in the working directory rather than the staging area.
  - `git reset --soft <commit id>` resets to specific commit, but will keep changes in the staging area
  - `git reset --hard <commit id>` resets to specific commit. Clears working directory and staging area so all later work is gone.

Resources

happygitwithr.com

github.com/tidyverse/dev-day-2019

- Issues tagged Help Wanted
- Issues tagged reprex

Thanks!