



git



GitHub

Introduction to git and GitHub

For Canepi! 😊



Welcome!

Here is what will cover in the training session:

- What is **Git** and **GitHub**
- General **Git** workflow
- Benefits of using **GitHub**
- How to use your own **Git/GitHub** accounts
- Downloading/ cloning repositories ('repos')
 - Website vs locally
- Uploading our own repo
 - Locally vs website
- Create and edit your own live website via **GitHub!**





Welcome!

Here is what will cover in the training session:

- What is **Git** and **GitHub**
- General **Git** workflow
- Benefits of using **GitHub**
- How to use your own **Git**/**GitHub** accounts
- Downloading/ cloning repositories ('repos')
 - Website vs locally
- Uploading our own repo
 - Locally vs website
- Create and edit your own live website via **GitHub**!



What will you need?

Prior to the training session, you will need to have setup 4 different programmes/ accounts + 2 other 'image related' tasks.

1. Text editor (programme)
 - Atom - <https://atom.io/>
(preferable over Notepad ++)
 - Notepad ++ <https://notepad-plus-plus.org/downloads/>
2. **Git** (programme)
 - <https://git-scm.com/downloads>
3. **GitHub** (account)
 - <https://github.com/>
4. **GitHub** (desktop)
 - <https://desktop.github.com/>
5. 3 'personal' photos that you are happy to share and describe!
6. Think of 1 cartoon or animal.

For a more detailed setup guide see the other presentation:
'GitHub_training_intro_prior_setup_instructions.pptx'

https://github.com/SBurnard/GitHub_training



What is **Git** and **GitHub**

- **Git** is version control system (works locally on your machine).
- **GitHub** is a centralised/ 'cloud' hosting service for **Git** repositories.

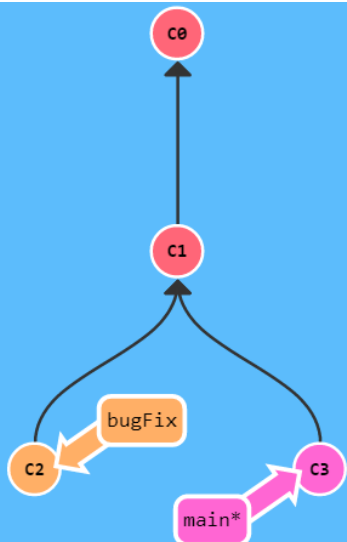


Benefits (and examples) of **Git**!

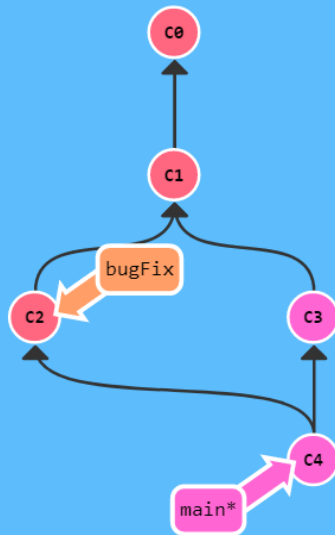
- **Git** is version control system (works locally on your machine).
- **GitHub** is a centralised/ 'cloud' hosting service for **Git** repositories.
- **Git** (version control) is most useful for programmers!
 - So probably of less interest to yourselves.



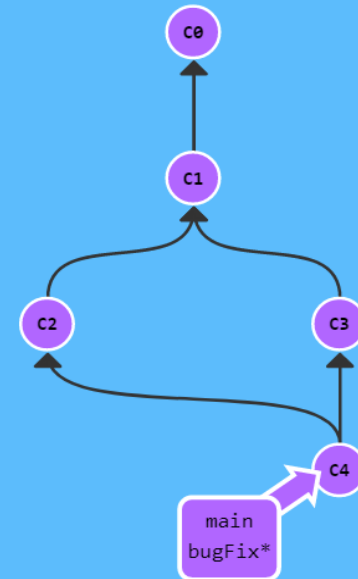
Example



git merge bugFix



git checkout bugFix; git merge main



Example of needing to return to an earlier version...

C3

Benefits (and examples) of **Git** – you would know!

- **Git** is version control system (works locally on your machine).
- **GitHub** is a centralised/ 'cloud' hosting service for **Git** repositories
- **Git** (version control) is most useful for programmers!
 - So probably of less interest to yourselves.

C1

Benefits (and examples) of **Git** – you !

- **Git** is version control system (works locally on your machine).
- **GitHub** is a centralised/ 'cloud' hosting service for **Git** repositories
- **Git** (version control) is most useful for programmers!
 - So probably of less interest to yourselves.

C0

Benefits (and examples) of **Git**!

- **Git** is version control system (works locally on your machine).
- **GitHub** is a centralised/ 'cloud' hosting service for **Git** repositories
- **Git** (version control) is most useful for programmers!
 - So probably of less interest to yourselves.



Benefits (and examples) of **GitHub**!

- **Git** is version control system (works locally on your machine).
- **GitHub** is a centralised/ 'cloud' hosting service for **Git** repositories.
- **Git** (version control) is most useful for programmers!
 - So probably of less interest to yourselves.
- **GitHub** very useful for researchers
 1. **Store** and **share** your data/ code
 2. **Access** tonnes/ 'heaps' of open source programmes and data
 3. **FREE!**
 4. **Versatile.**



Benefits (and examples) of GitHub!

Example GitHub page from a lab group sharing software:

- <https://github.com/mhammell-laboratory>
- <https://github.com/PMBio>

Example GitHub page from an individual:

- <https://github.com/sirselim>

Storing and sharing training:

- <https://github.com/PMBio/SingleCellCourse>

Data and information for a publication:

- <https://github.com/PMBio/scNMT-seq>

Easily create webpages:

- <https://learning-zone.github.io/website-templates/avenger-multi-purpose-responsive-html5-bootstrap-template/> (generic example template - see more <https://github.com/learning-zone/website-templates>)
- https://sirselim.github.io/tSNE_plotting/ (Interactive plots on a webpage linked to a publication)
- https://sirselim.github.io/tSNE_plotting/img/test_large_tsne_plot.html

Share your journey to promote a tool, generate interest and invite/ benefit from collaborative troubleshooting:

- https://github.com/sirselim/jetson_nanopore_sequencing (his current journey developing a cheap and portable setup for the Nanopore!)

These are just a few ‘basic’ example uses of GitHub

Workflow Diagram

1°. add
2°. commit
3°. push

**Our
computer**

DEV ENVIRONMENT

**WORKING
DIRECTORY**

**STAGING
AREA**

**LOCAL
REPOSITORY**

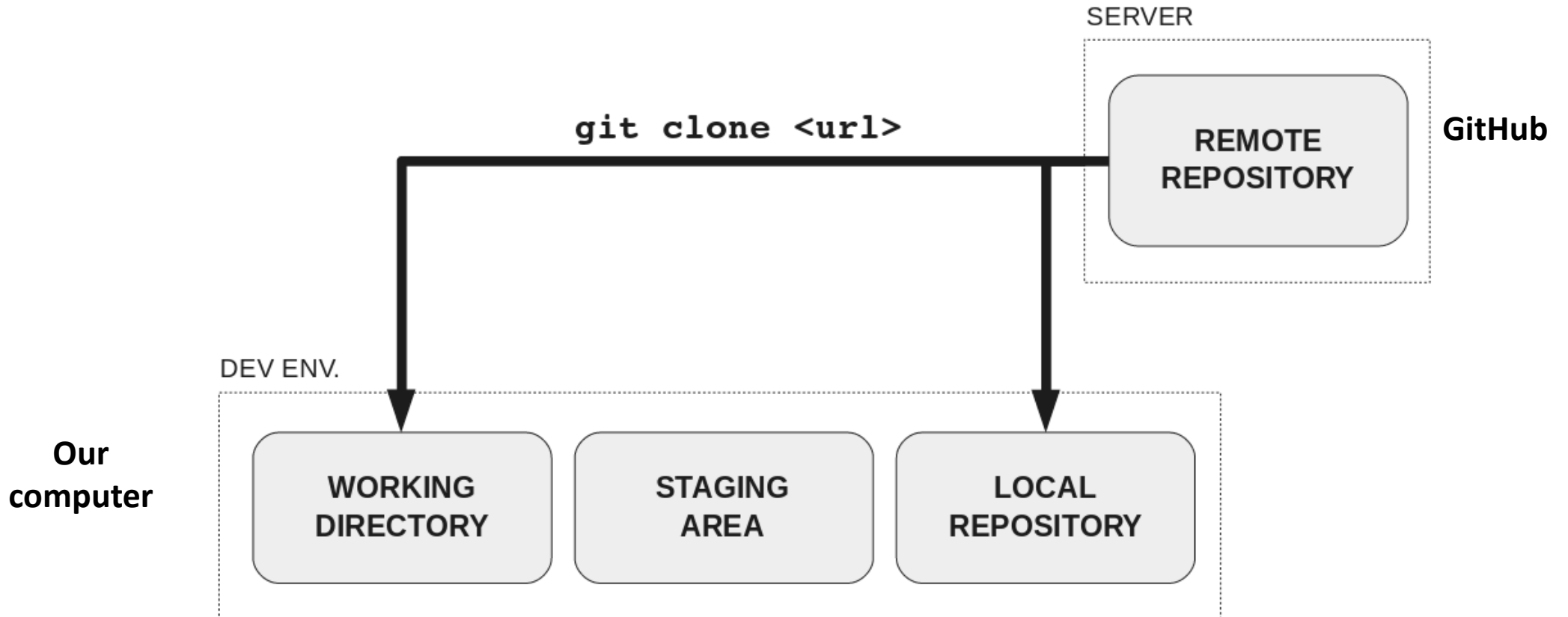
SERVER

**REMOTE
REPOSITORY**

GitHub

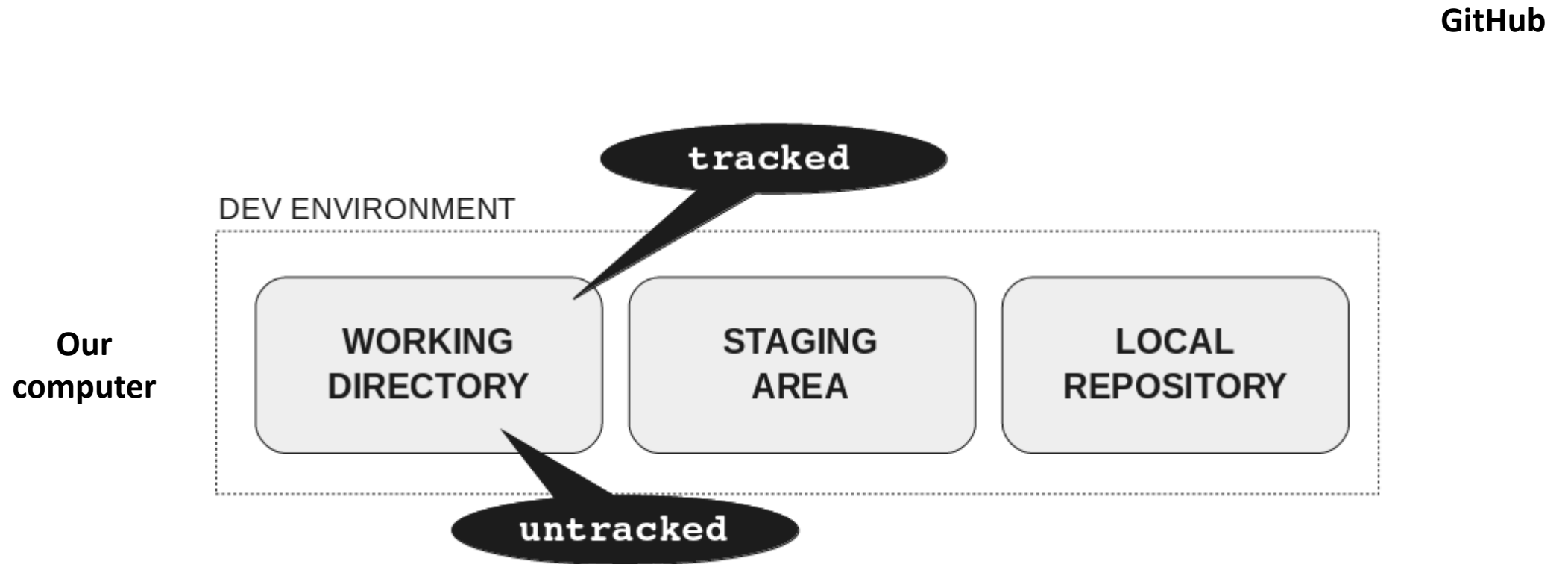
1°. add
2°. commit
3°. push

Workflow Diagram



1°. add
2°. commit
3°. push

Workflow Diagram



Workflow Diagram

**Our
computer**

DEV ENVIRONMENT

**WORKING
DIRECTORY**

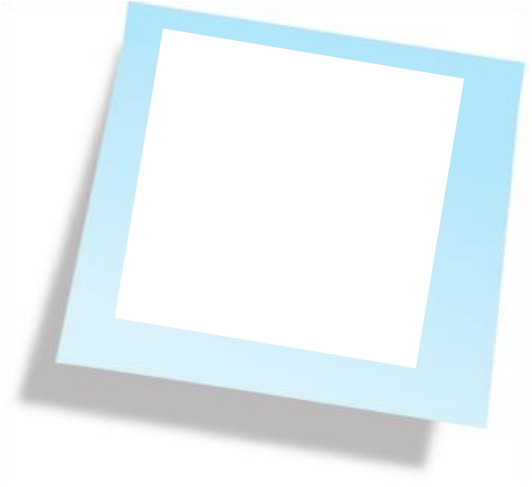
**STAGING
AREA**

**LOCAL
REPOSITORY**

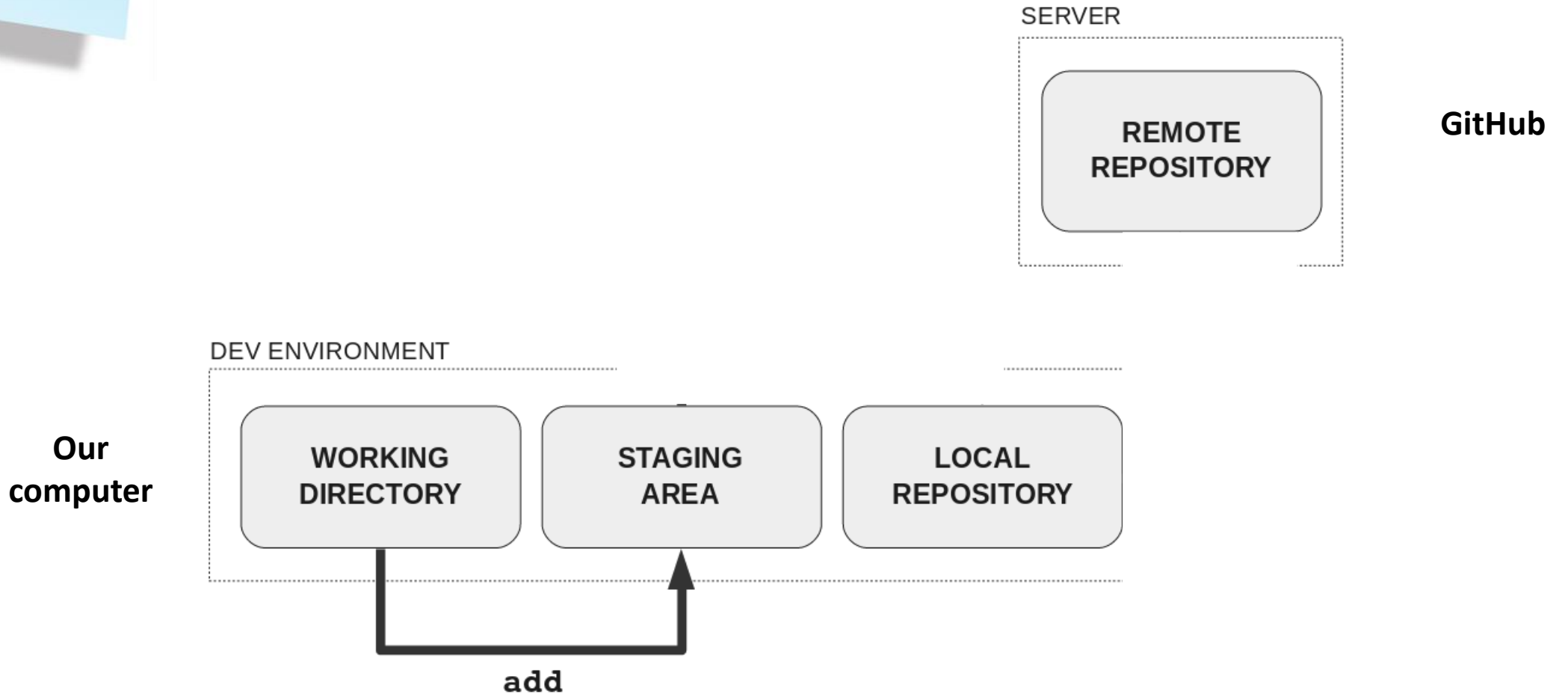
SERVER

**REMOTE
REPOSITORY**

GitHub



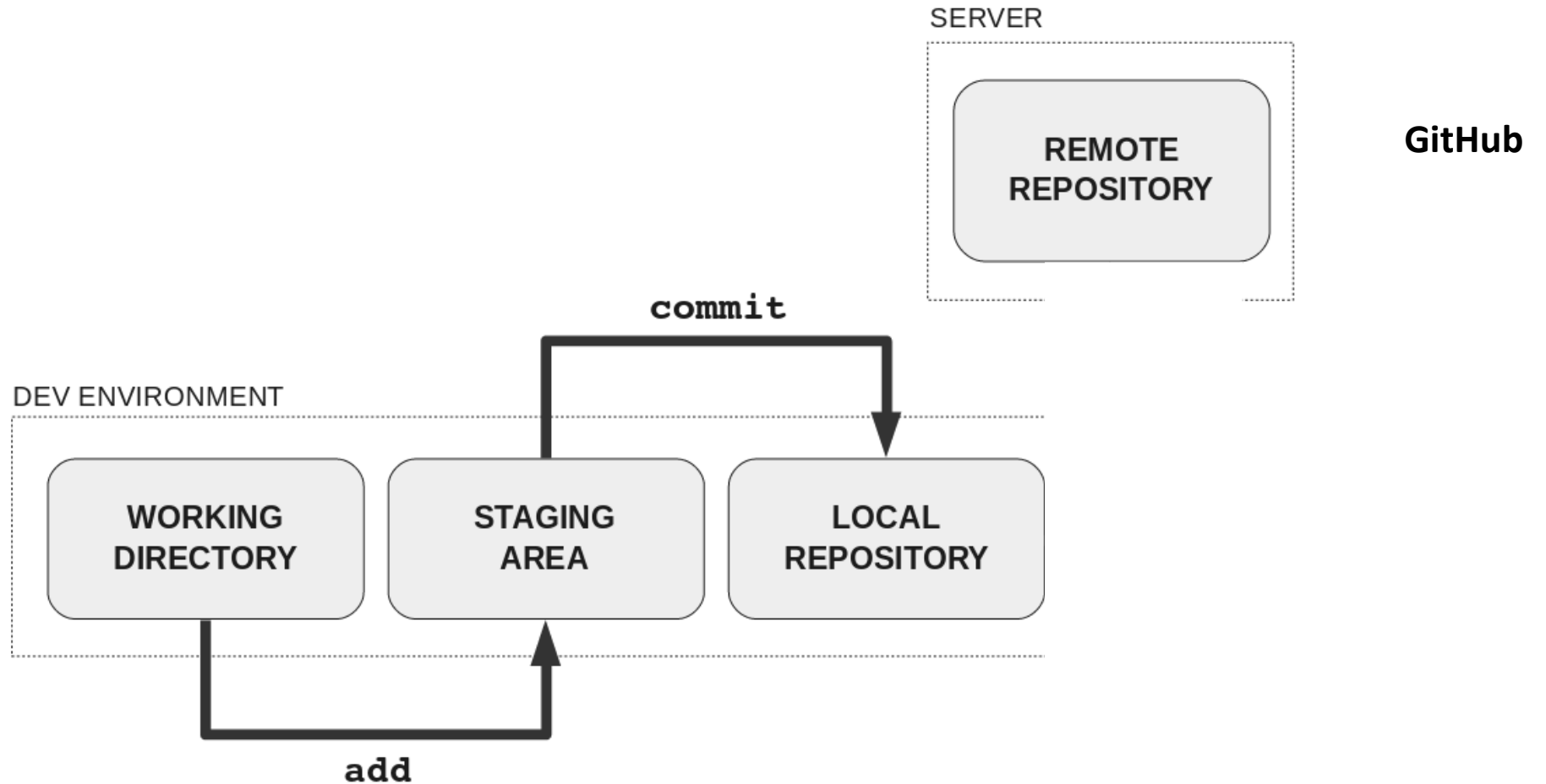
Workflow Diagram



Workflow Diagram

1º. add
2º. commit

Our
computer



1°. add
2°. commit

Git Commits

A commit in a git repository records a snapshot of all the (tracked) files in your directory. It's like a giant copy and paste, but even better!

Git wants to keep commits as lightweight as possible though, so it doesn't just blindly copy the entire directory every time you commit. It can (when possible) compress a commit as a set of changes, or a "delta", from one version of the repository to the next.

Git also maintains a history of which commits were made when. That's why most commits have ancestor commits above them -- we designate this with arrows in our visualization. Maintaining history is great for everyone working on the project!

It's a lot to take in, but for now you can think of commits as snapshots of the project. Commits are very lightweight and switching between them is wicked fast!

Our
computer

DEV ENV

DIRECTORY

AREA

REPOSITORY

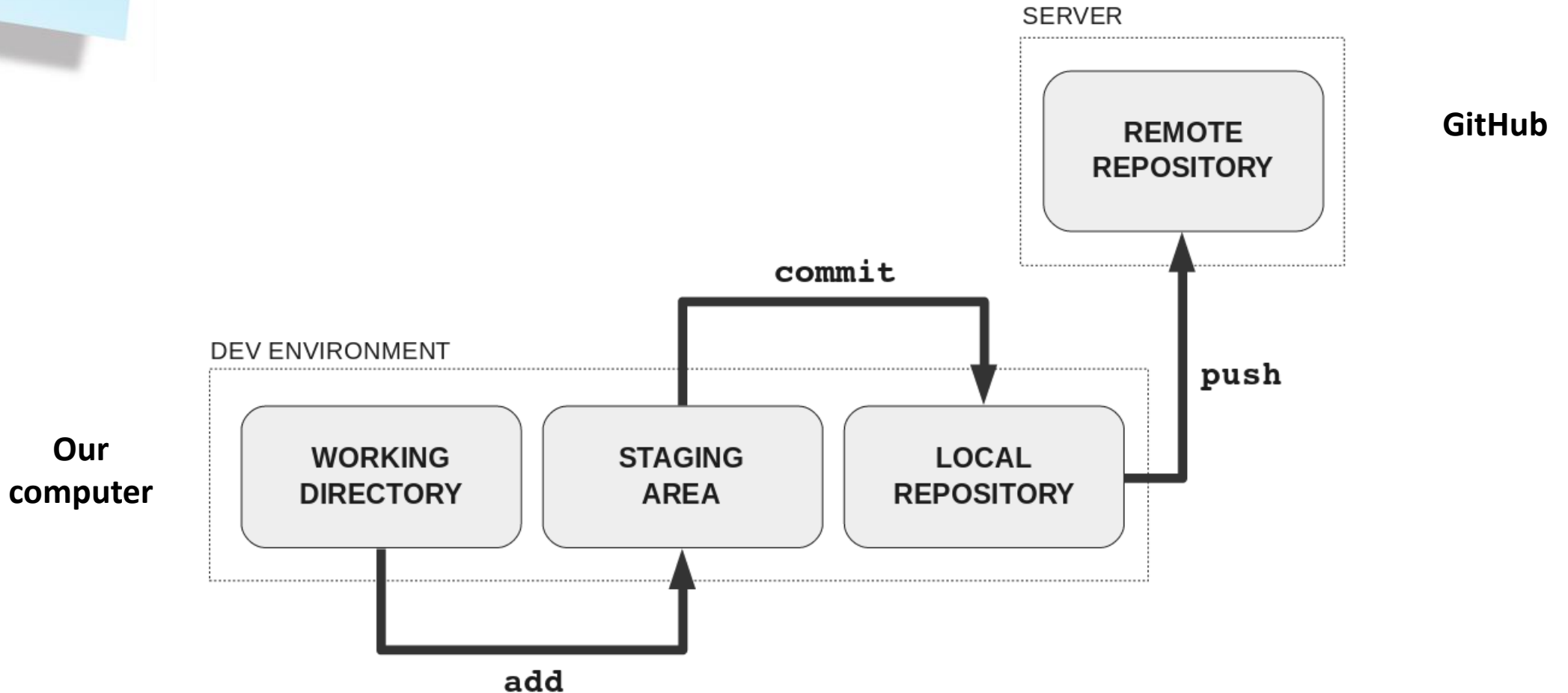


add

Workflow Diagram

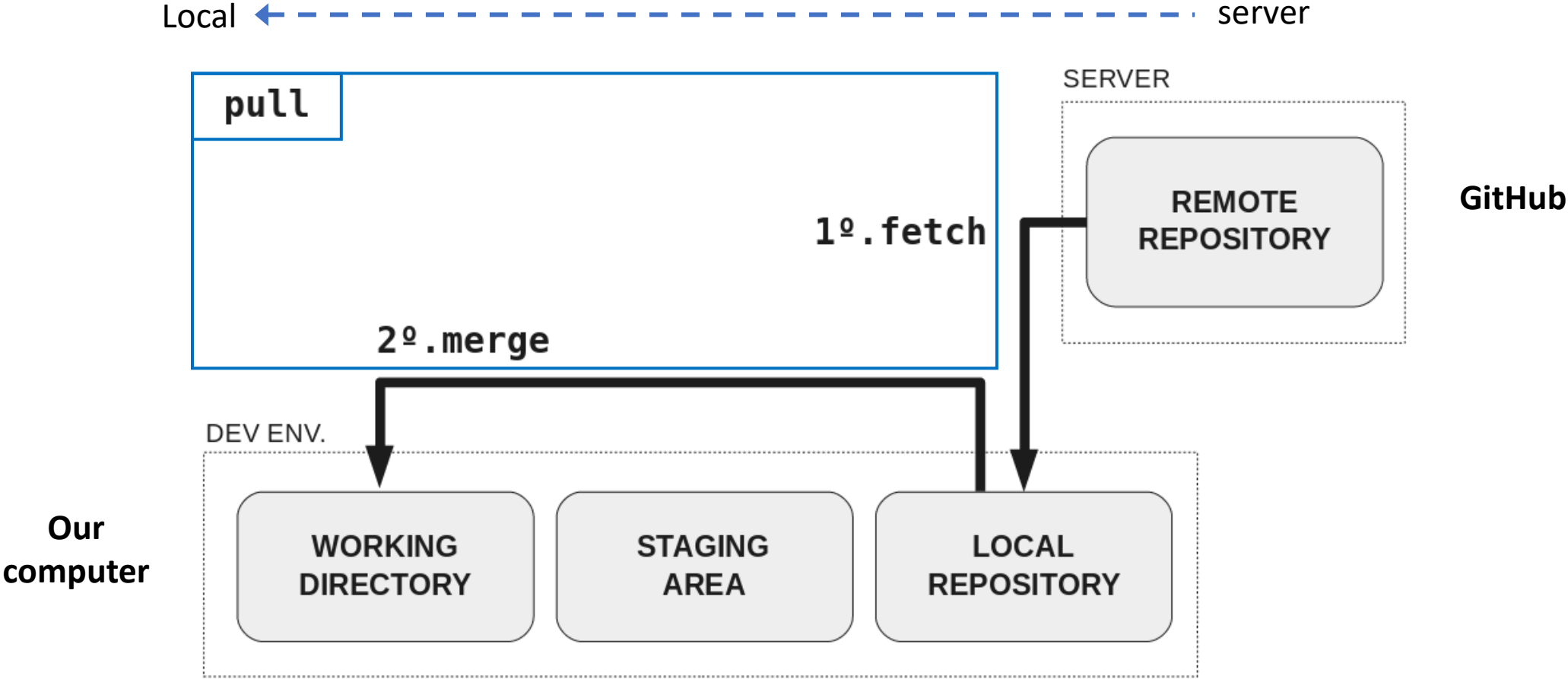
1°. add
2°. commit
3°. push

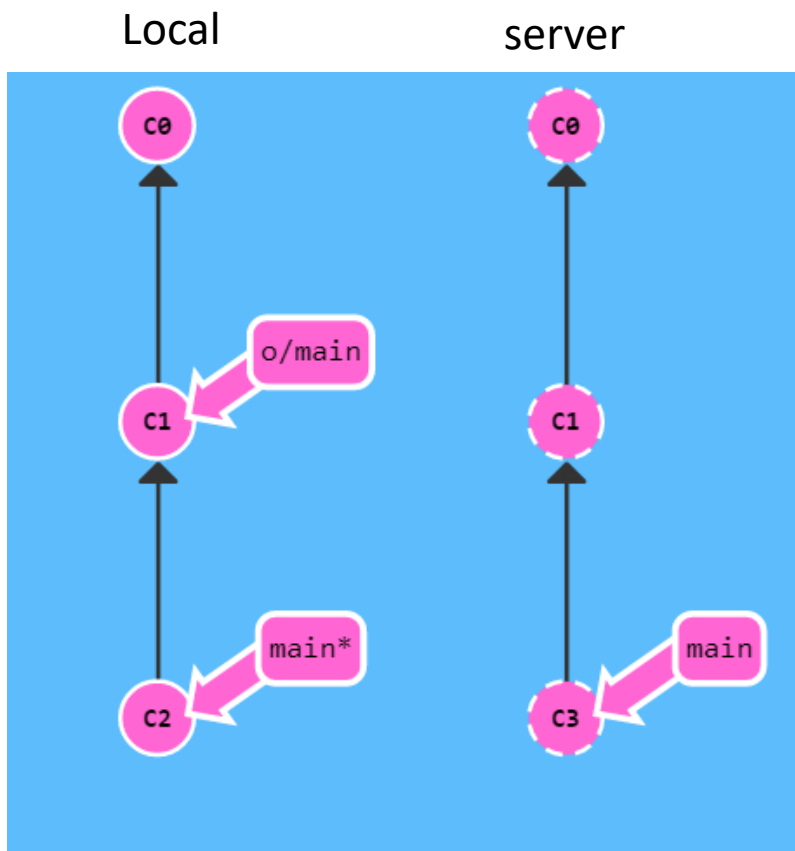
Local —————> server



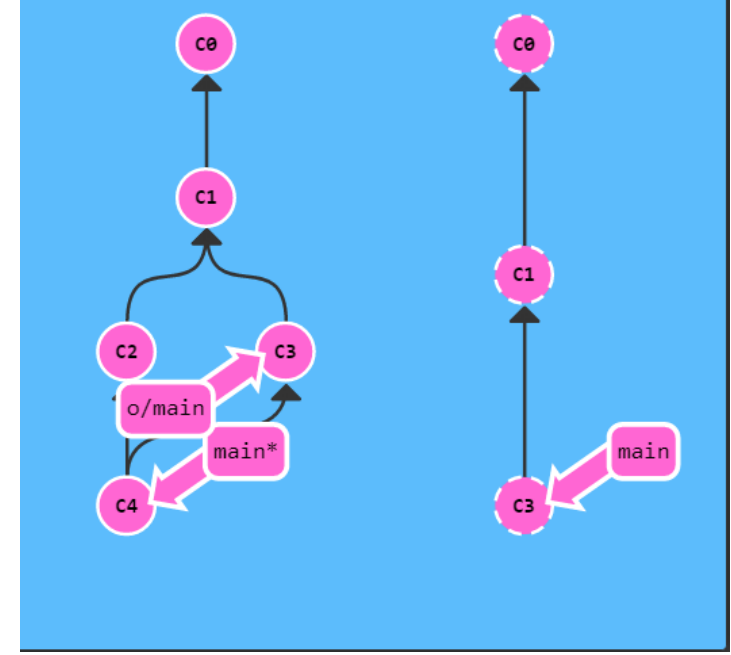
1°. add
2°. commit
3°. push

Workflow Diagram



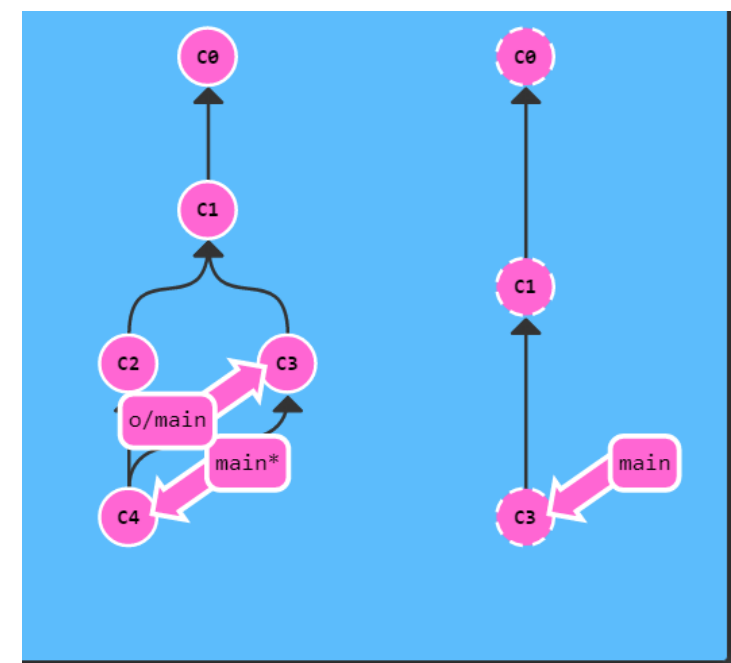


`git fetch; git merge o/main`



Local server

`git pull`



local

clone

github.com/KlugerLab/ALRA

My comp/ HPC

- data
- .gitignore
- LICENSE.txt
- README.md
- alra.R
- alraSeurat2.R
- alra_test.R

KlugerLab / ALRA

<> Code ! Issues 4 🔗 Pull requests ▶ Actions 📁 Projects 📖 Wiki ! Security 🔍 Insights

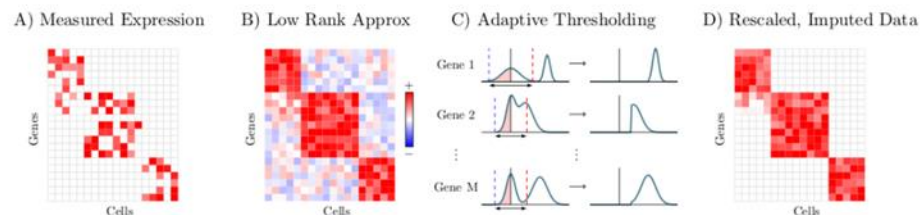
master	3 branches	0 tags	Go to file	Add file	Code
linqiaozhi Added license 7636de8 on 11 Aug 2019 20 commits					
data	first commit	3 years ago			
.gitignore	Added support for intel MKL implementation of SVD	2 years ago			
LICENSE.txt	Added license	2 years ago			
README.md	Update README.md	2 years ago			
alra.R	Switched away from using a normal distribution to approximate spacings	2 years ago			
alraSeurat2.R	alra with Seurat2 object	2 years ago			
alra_test.R	Update alra_test.R	2 years ago			

README.md

Adaptively-thresholded Low Rank Approximation (ALRA)

Introduction

ALRA is a method for imputation of missing values in single cell RNA-sequencing data, described in the preprint, "Zero-preserving imputation of scRNA-seq data using low-rank approximation" available [here](#). Given a scRNA-seq expression matrix, ALRA first computes its rank-k approximation using randomized SVD. Next, each row (gene) is thresholded by the magnitude of the most negative value of that gene. Finally, the matrix is rescaled.



Packages

No packages published

Contributors 2

linqiaozhi George Linderman

JunZhao1990 Jun Zhao

Languages

R 100.0%

Exercises

1. Clone GitHub training repo

- Using Git bash terminal
- Open document

2. Make and edit your own repo

- Using GitHub.com and git bash terminal
- Copy contents of GitHub training.
- Create a note and upload with your favourite animal.
- These default as public (keep it public for this one).

3. Find a friend

- On the GitHub website
- Find each other – and find out their favourite animal... ;)

4. Fork and create website!

- Using GitHub desktop and GitHub.com
- Check your coding friends page and see if their ‘favourite animal’ featured on their webpage. :D

Exercise 1– ‘Clone my repo!’

Clone this GitHub training github:

https://github.com/SBurnard/GitHub_training

1. Open the ‘Git Bash’ programme.

2. Move to your desktop

`cd Desktop`

3. Clone the repo

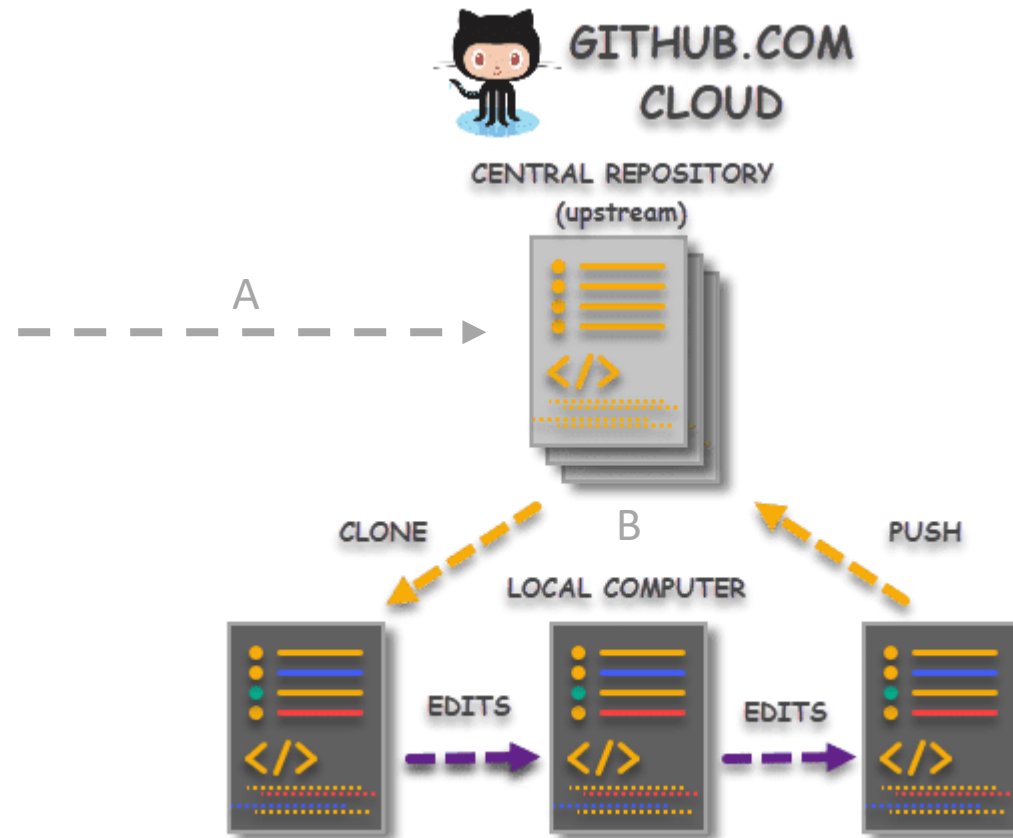
`git clone https://github.com/SBurnard/GitHub_training`

DONE! You’ve now made an exact copy of all this directory and all of it’s files.

4. Go check it out. (And open the exercises doc)

Exercise 2 – Make you own Repo

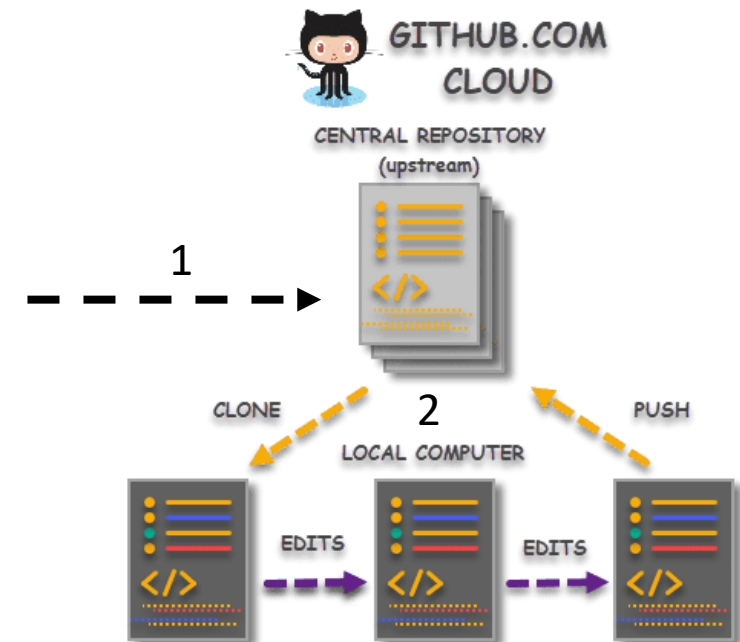
A) Direct upload to GitHub and B) edit (via command line)



Exercise 2 – Make you own Repo

Aim:

- 1) Make you own Repo on GitHub (directly uploaded to GitHub.com)
 - 2) Modify/ edit (via command line – Git Bash)
- Follow the instructions in the exercises doc (from the repo you just cloned).



Exercise 3 – Find a friend!

Collaborations require.... Collaborators! 😊

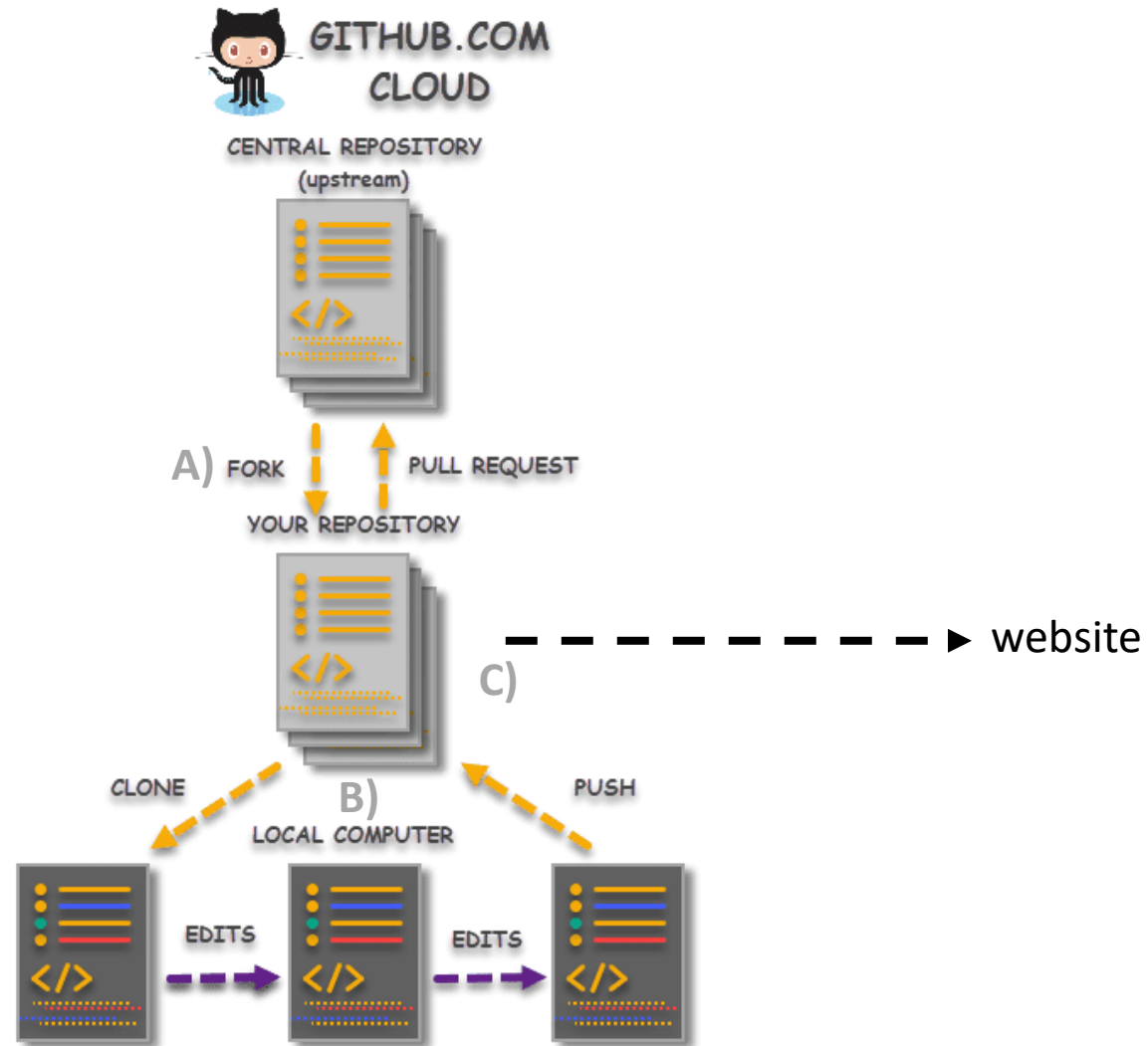
Find and follow each other. Three 'easy' methods:

- 1) use the search bar and click user.
- 2) type users:<name> directly in the search bar
- 3) Ask your friend their username and type github.com/<Their_User_Name>

And... Most importantly what was their favourite animal?! :D
(Go search their public repos)

Exercise 4 – A) Fork, B) edit and C) upload your own website!

(via GitHub desktop)



Why go through all this effort and not just allow 'autosaving'?

- Such as often defaulted in word or excel....

Questions....



What is GitHub?

Questions?



What have you learnt/ done?

- Learnt and used both **git** and **GitHub**
- Various benefits and uses of GitHub
- Have your own GitHub account:
 - Created your own repo
 - Launched your own website!
 - Made some friends/ collaborators.
- Used several different methods:
 - Git bash terminal
 - Git Desktop
 - GitHub.com (directly uploading and editing on the website)



Good additional training options

From GitHub:

- <https://lab.github.com/githubtraining/prepare-to-use-github>
- <https://lab.github.com/githubtraining/introduction-to-github>
- <https://lab.github.com/githubtraining/first-day-on-github>

Alternative sites:

- <https://blog.upperlinecode.com/how-to-teach-git-commits-github-to-teenagers-a3f740b2f500>
- <https://rachelcarmena.github.io/2018/12/12/how-to-teach-git.html>
- <https://towardsdatascience.com/getting-started-with-git-and-github-6fcd0f2d4ac6>
- <https://learngitbranching.js.org/> (interactive for git)