GRACE HOPPER CELEBRATION

/ ANITA B.ORG

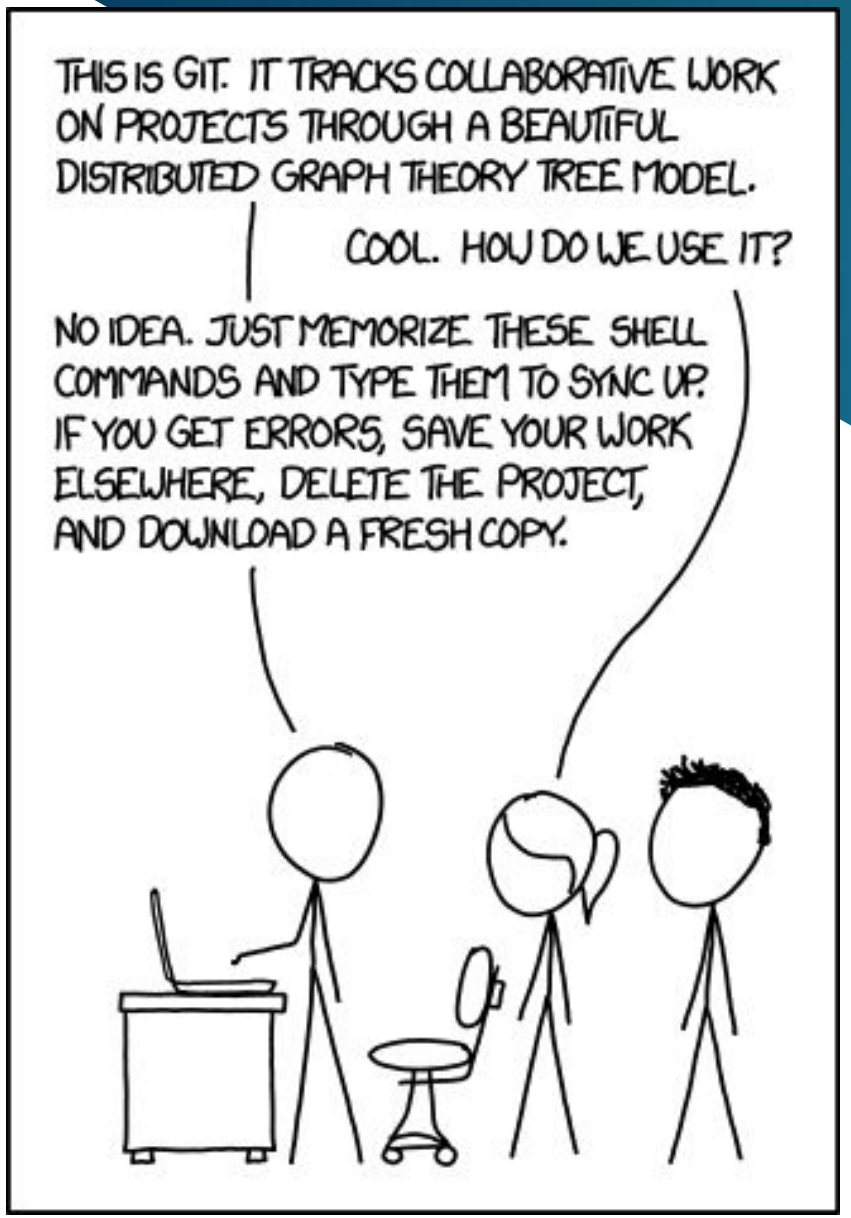
Branching Out

GitHub - Core Skills for Contributing to Open Source

Lily Sturmann
Red Hat, Software Engineer

Parul Singh
Red Hat, Software Engineer

#GHC19

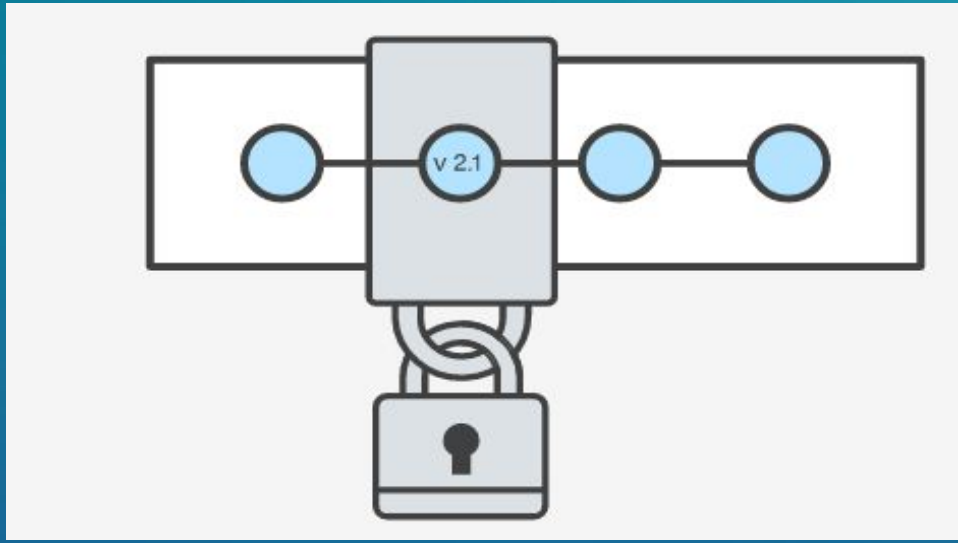


from https://xkcd.com/

Intro

What is Version Control?

Software tools for managing changes to source code over time



Keeps track of every modification to the code in a special kind of database

Why do Version Control?

Reducing risks by having a backup

A complete long-term change history of every file

Safe simultaneous modifications

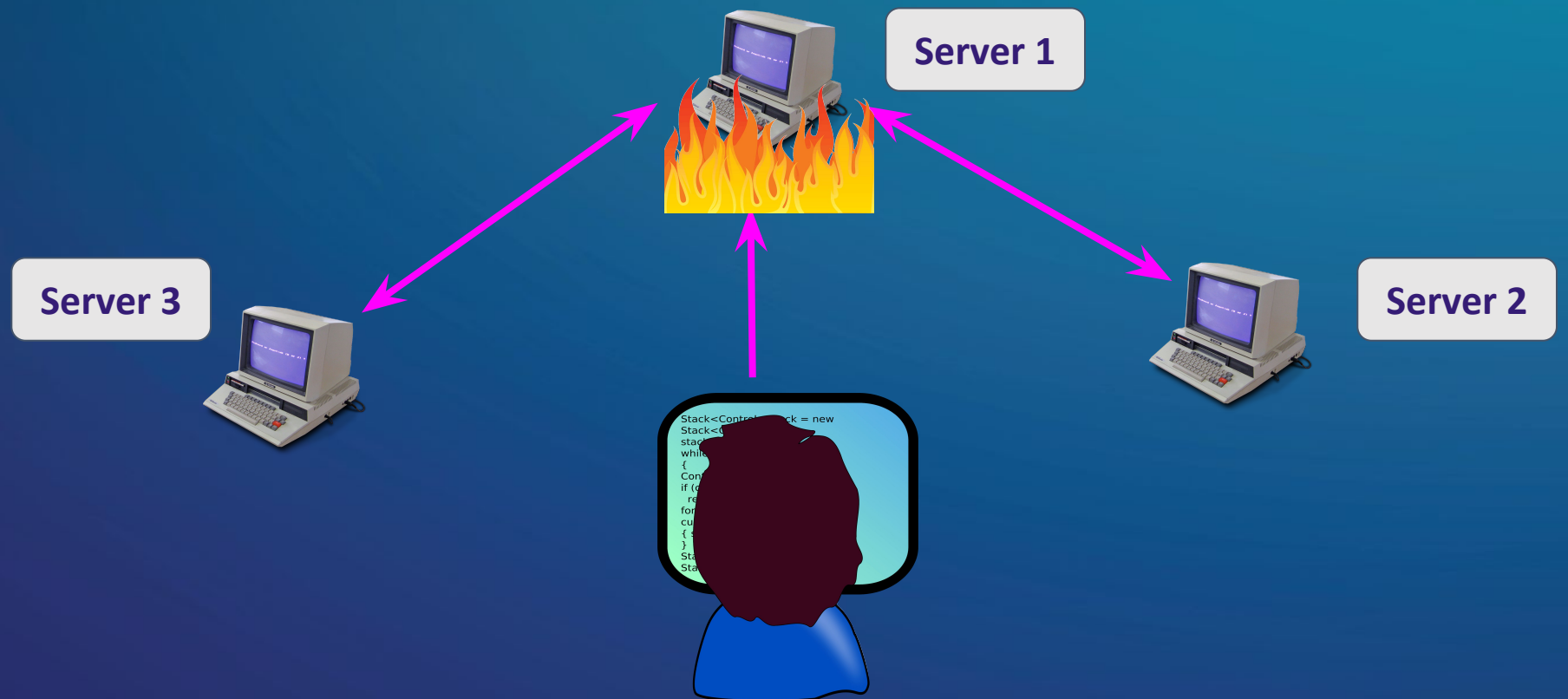
Collaboration

Kinds of Version Control

Case 1: No version control

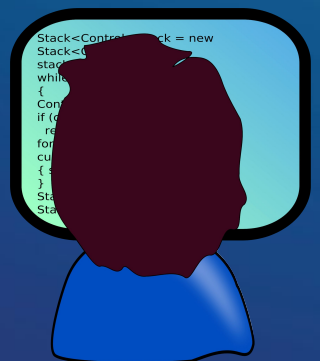
Kinds of Version Control

Case 2: Self-managed servers with version control system, e.g. Git, Mercurial



Kinds of Version Control

Case 3: An online/cloud version control system service



GitHub

Cloud-based service that helps developers store, manage and track their code

Easily accessible via a website
https://www.github.com

GitHub

Uses Git as the version control system

Makes it a lot easier for individuals and teams to use Git for version control and collaboration.

Git vs GitHub

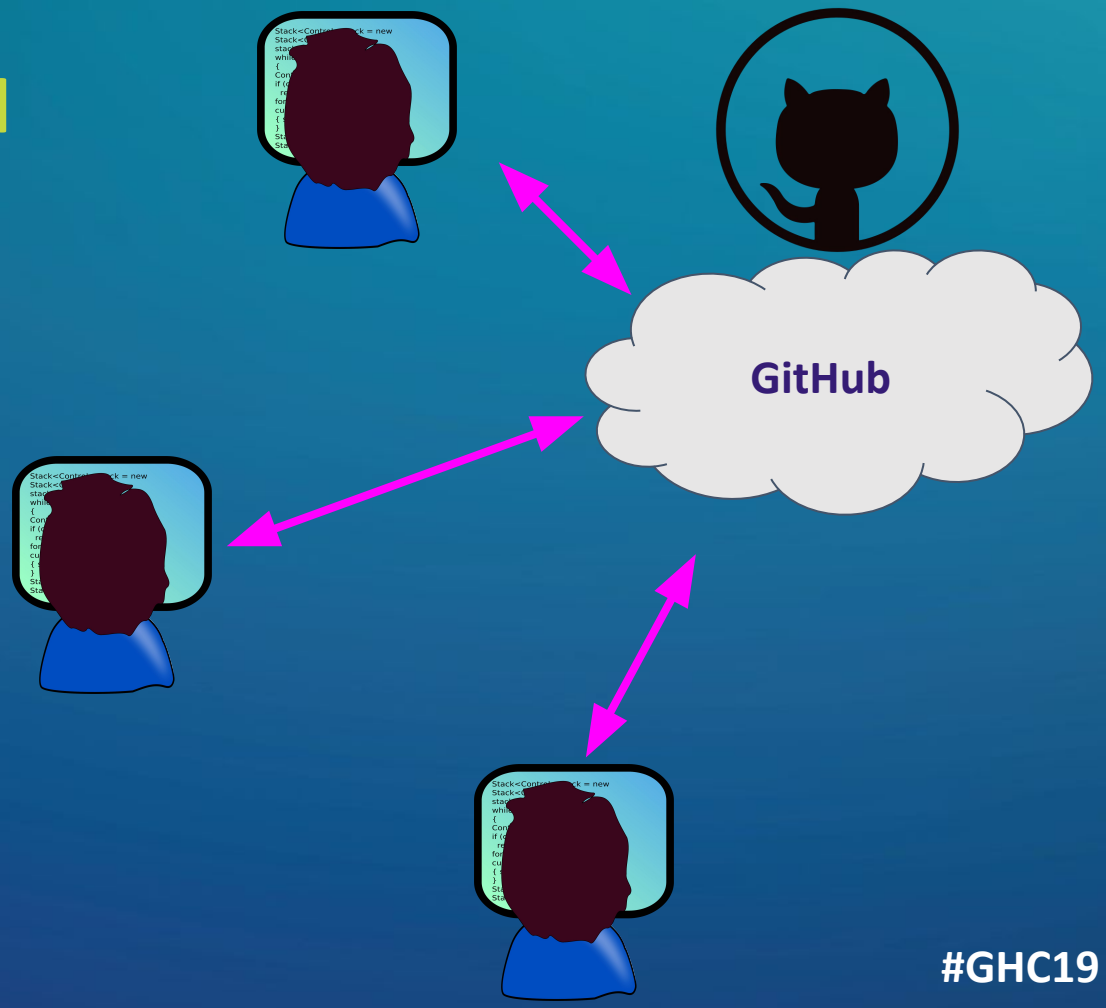
Git

Free and open source distributed version control system

GitHub

Hosting service for code that uses Git

Additional features on top of Git like IAM, UI, Pull Request etc.



Set-up

GitHub Set-Up

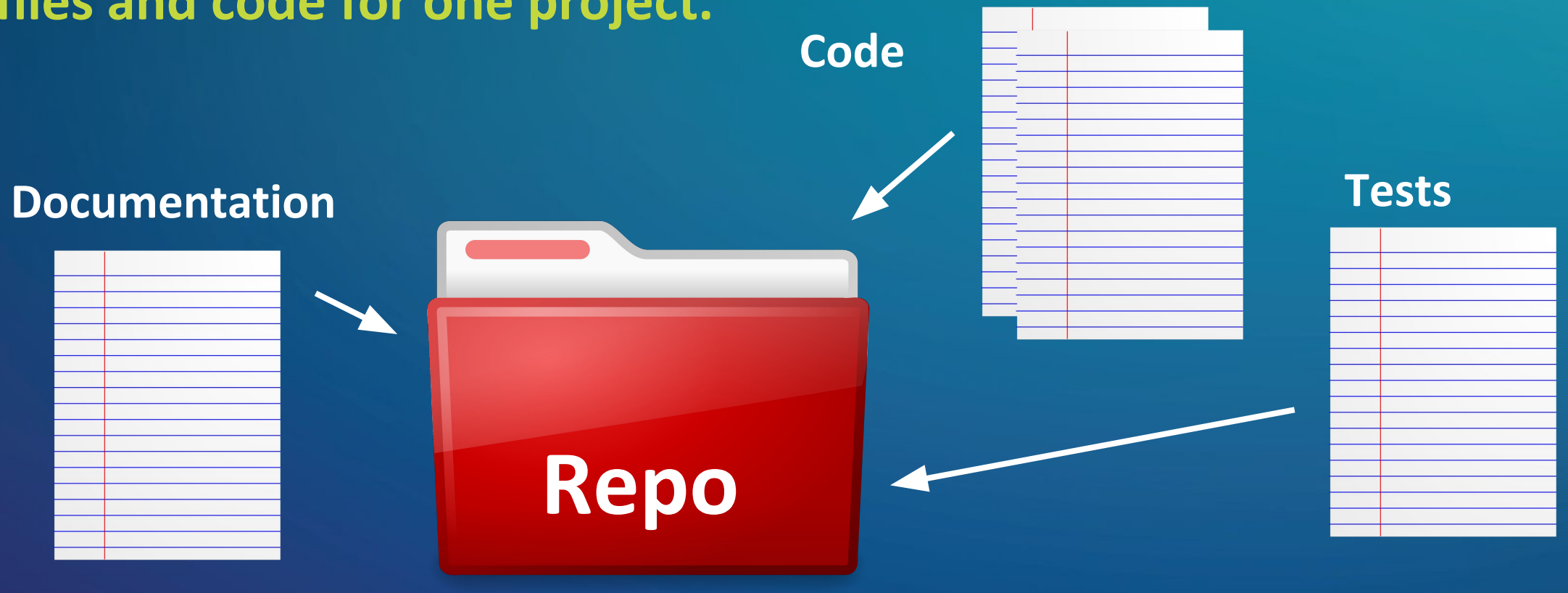
See specific commands in the commands doc:
https://github.com/github-fun/commands/blob/master/0-set-up.md

- ☑ Register a GitHub account
- ☑ Make sure git is installed on your laptop
- ☑ Configure git with your name and email

Creating a repo

Step 1.1: Create a new repo on GitHub

A repository (or “repo”) generally holds all of the files and code for one project.



Follow along: https://github.com/github-fun/commands/blob/master/1-repo.md

#GHC19

Step 1.2: Clone the repo locally

Before you clone, the repo exists only on GitHub. After you clone, the repo on your computer is a mirror image of the repo on GitHub.

First time: `git clone <URL>`

Then: `git pull` or `git fetch`

`git push <remote/branch name>`

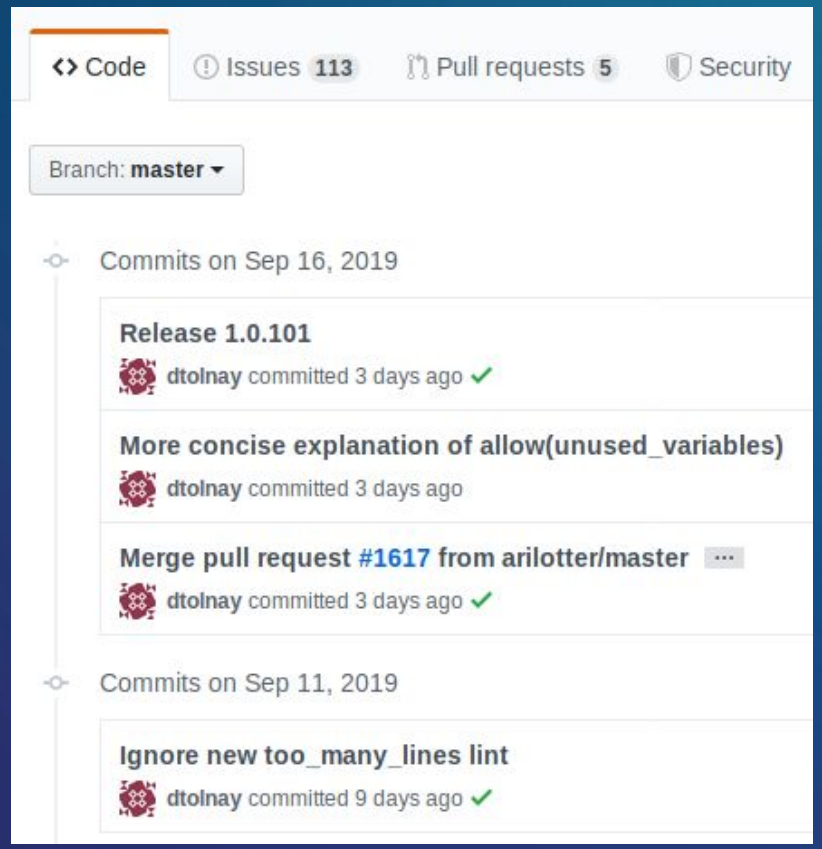
Edits can be made here!

#GHC19

Making changes

Git keeps track of change history

The same code or file looks different over time as a project evolves. Git keeps track of these changes at points in time called "commits" that you determine.



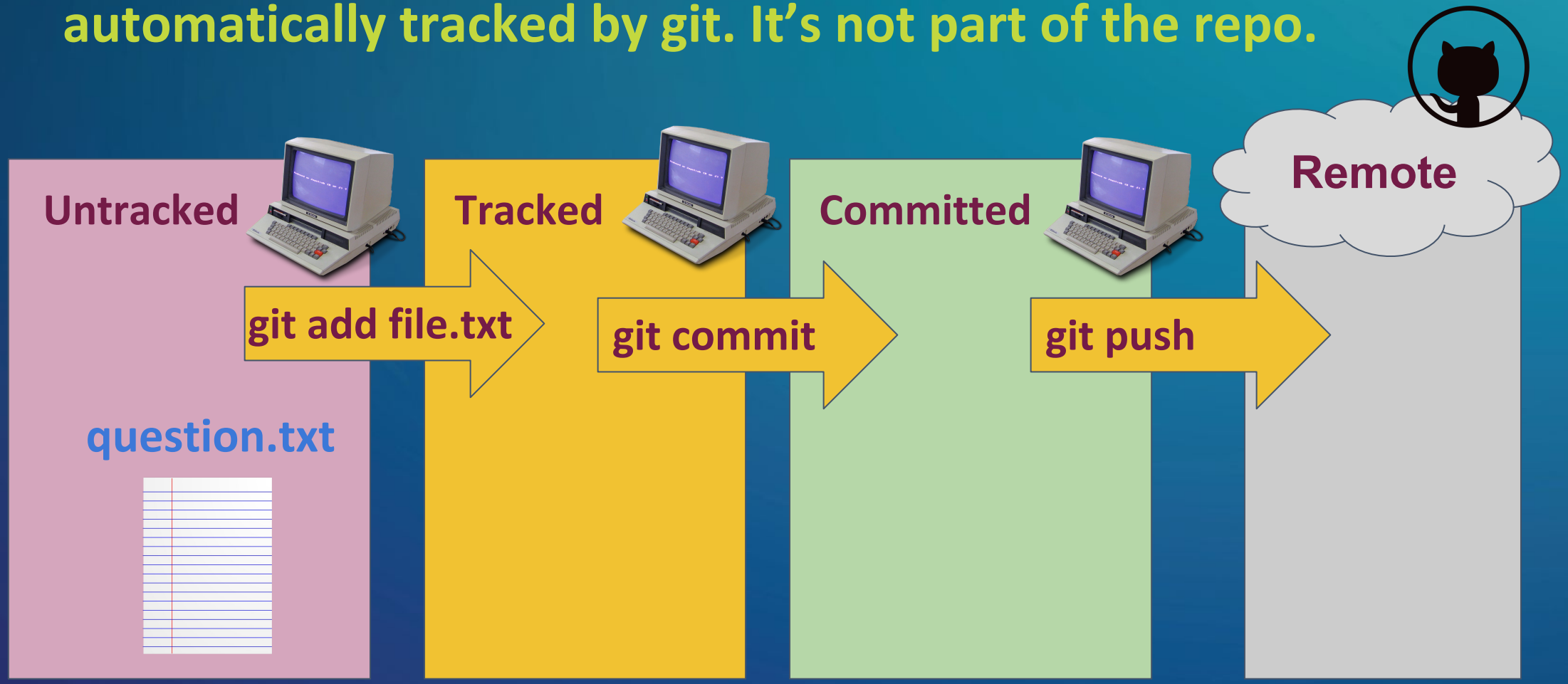
Today

Yesterday

Earlier

Step 2.1: Making changes (creating a file)

A file you create in your repo folder is not automatically tracked by git. It's not part of the repo.

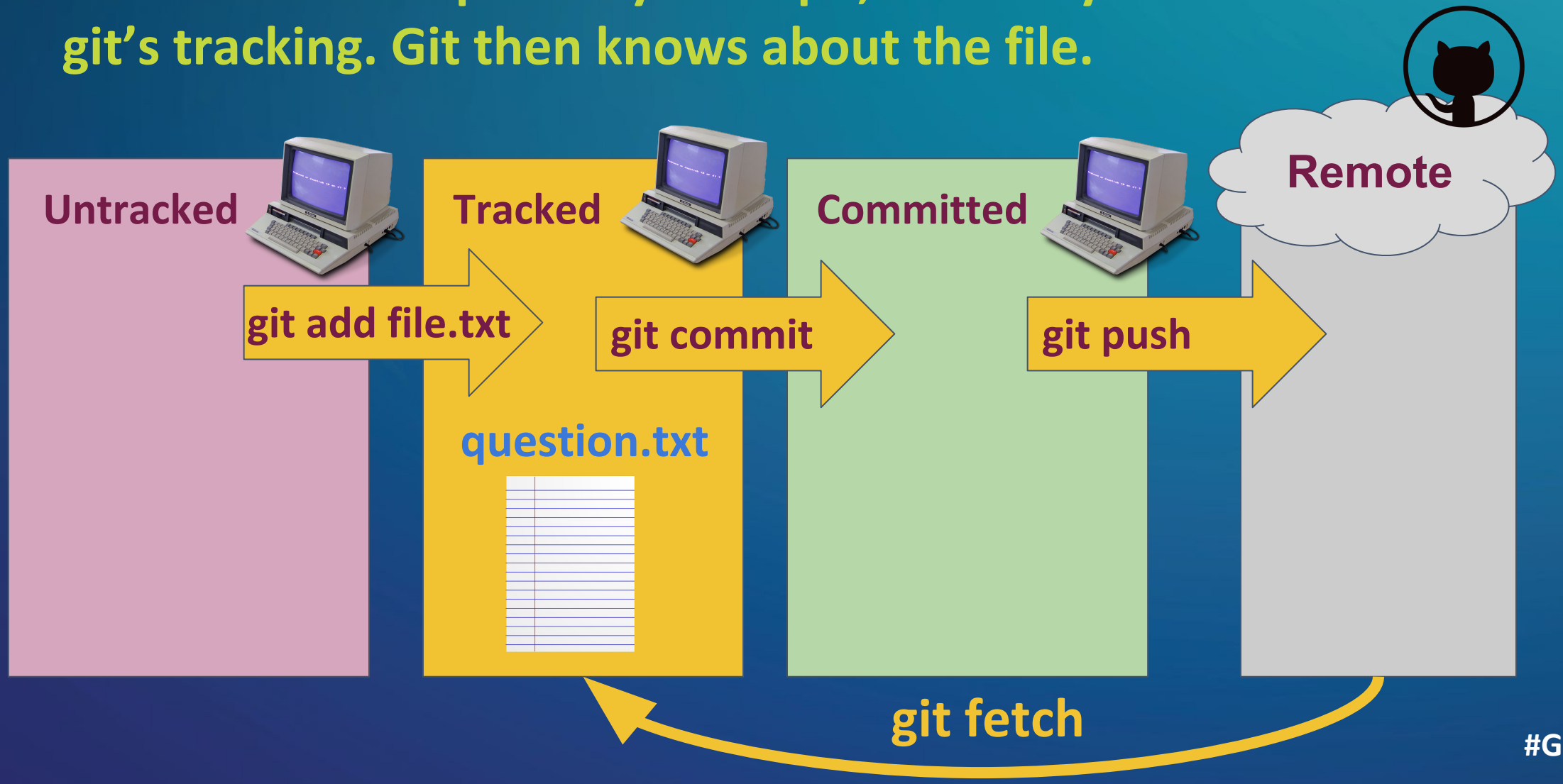


Follow along: https://github.com/github-fun/commands/blob/master/2-changes.md

#GHC19

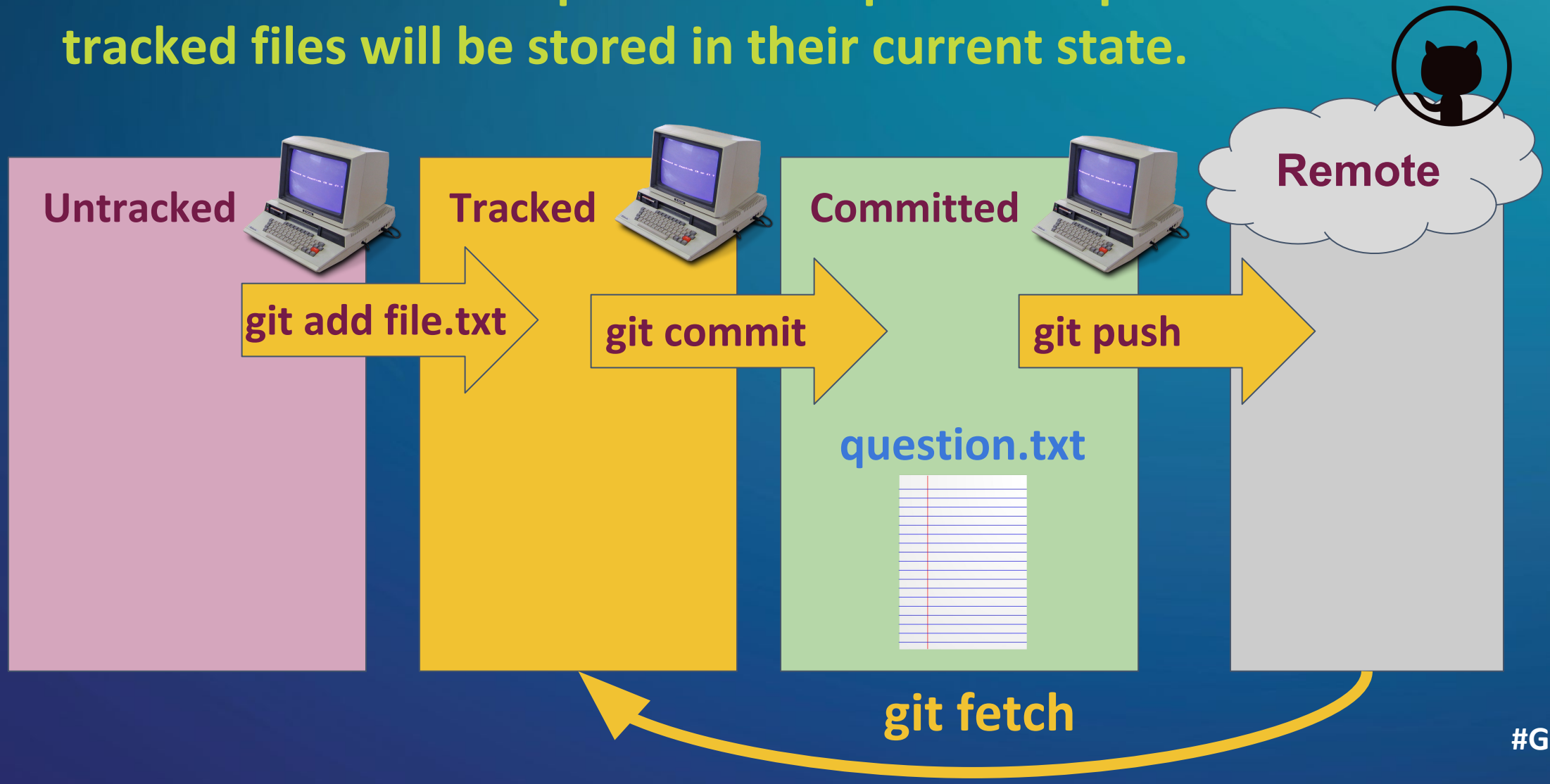
Step 2.2: Tracking changes

To make the file part of your repo, manually add it to git's tracking. Git then knows about the file.



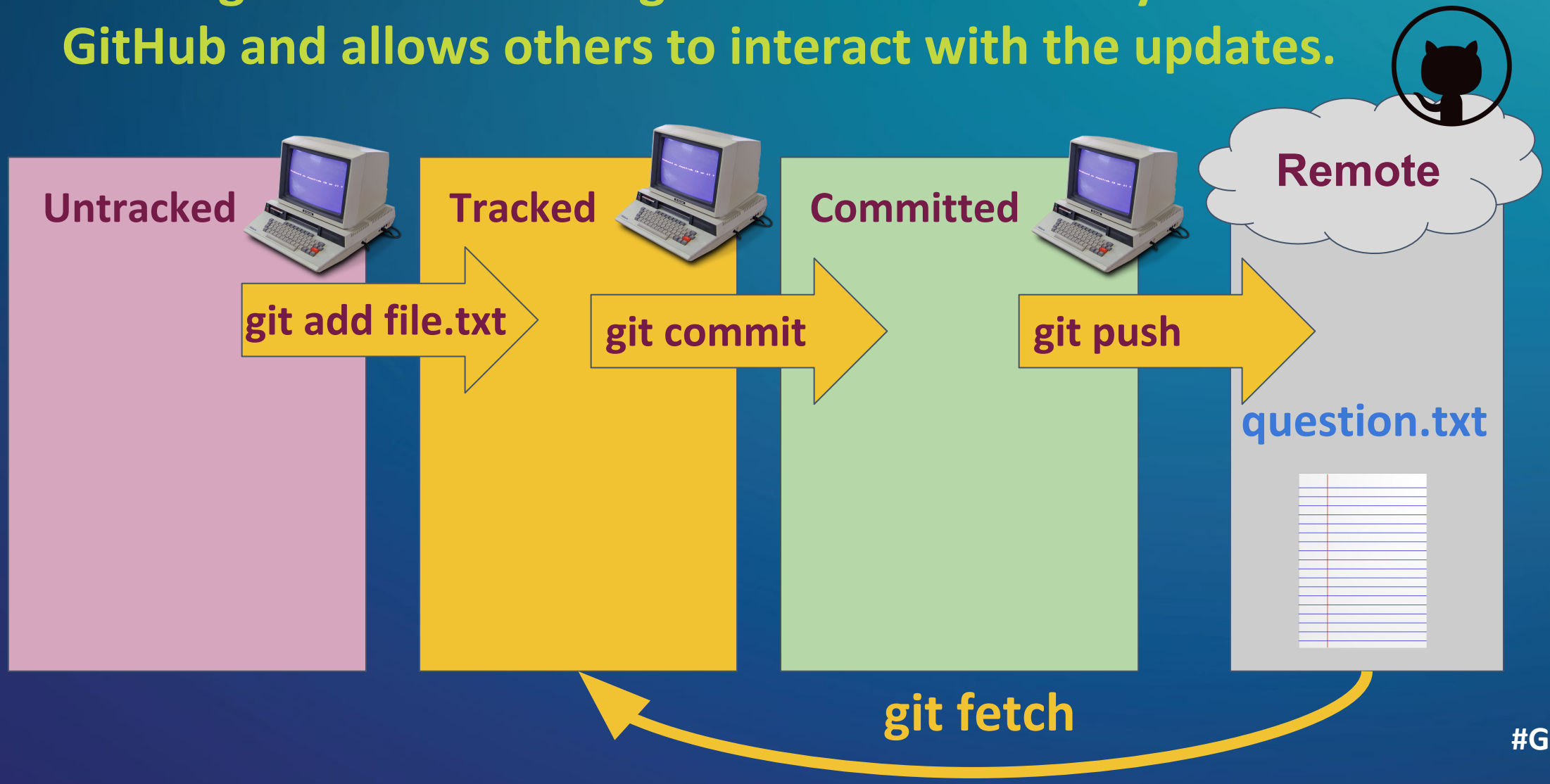
Step 2.3: Committing changes

A “commit” is a save point for a repo. The repo’s tracked files will be stored in their current state.



Step 2.4: Storing changes remotely

Pushing committed changes stores them safely with GitHub and allows others to interact with the updates.



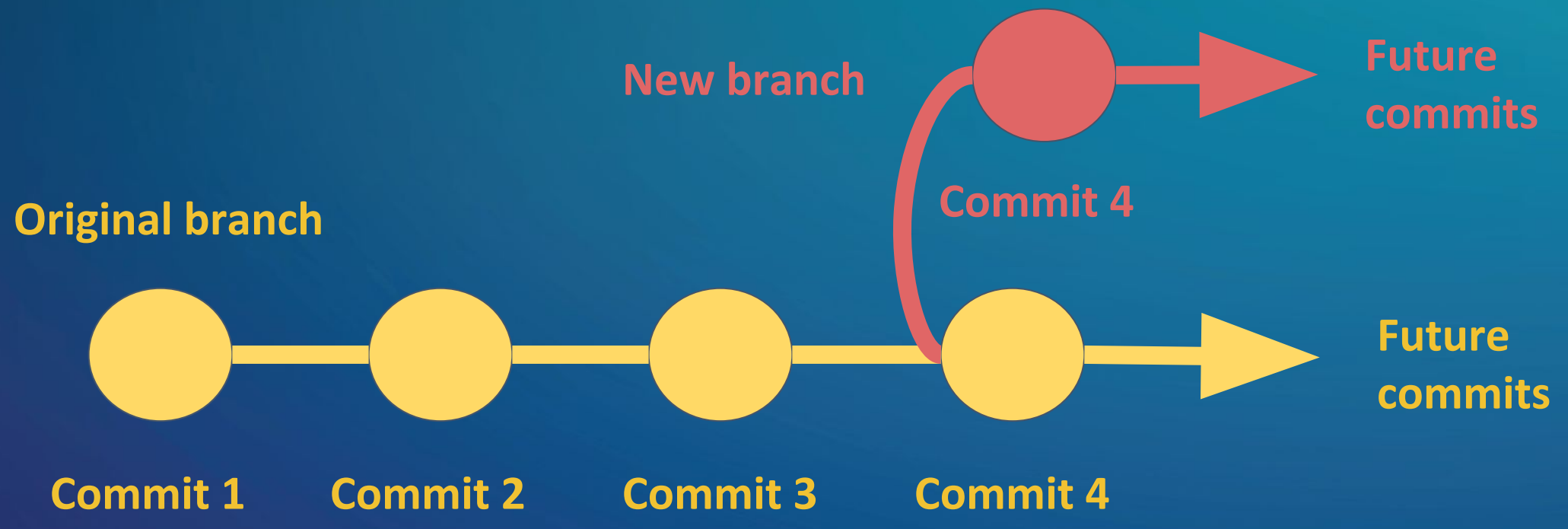
Branches

Git allows alternative versions

Git lets you try out different alternatives without altering your current changes. This makes it easy to work on several parts of a project at once or experiment boldly.

Step 3.1: Creating a new branch

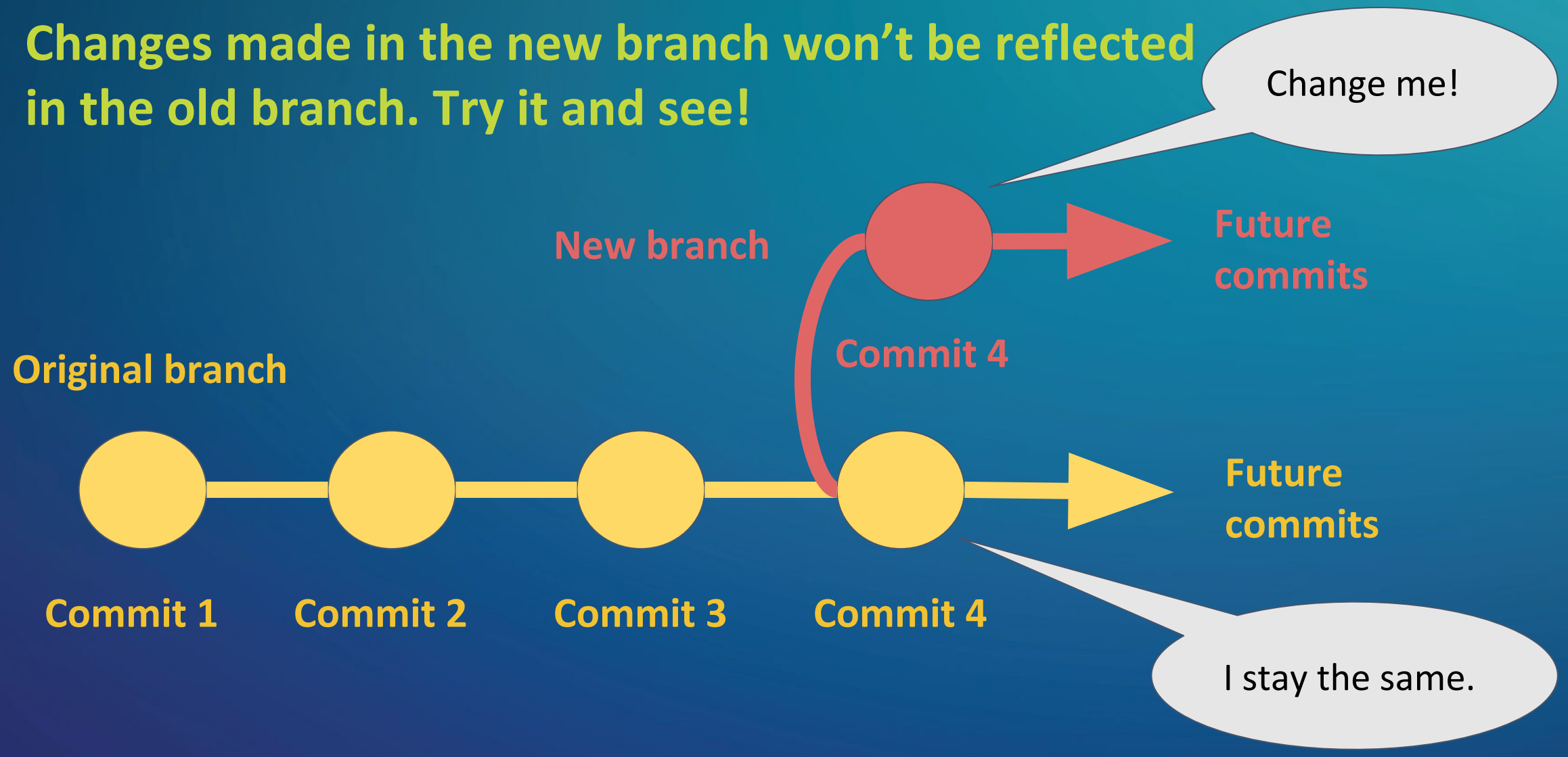
Branches keep changes organized. You can create or delete branches, or merge them together.



Follow along: https://github.com/github-fun/commands/blob/master/3-branches.md

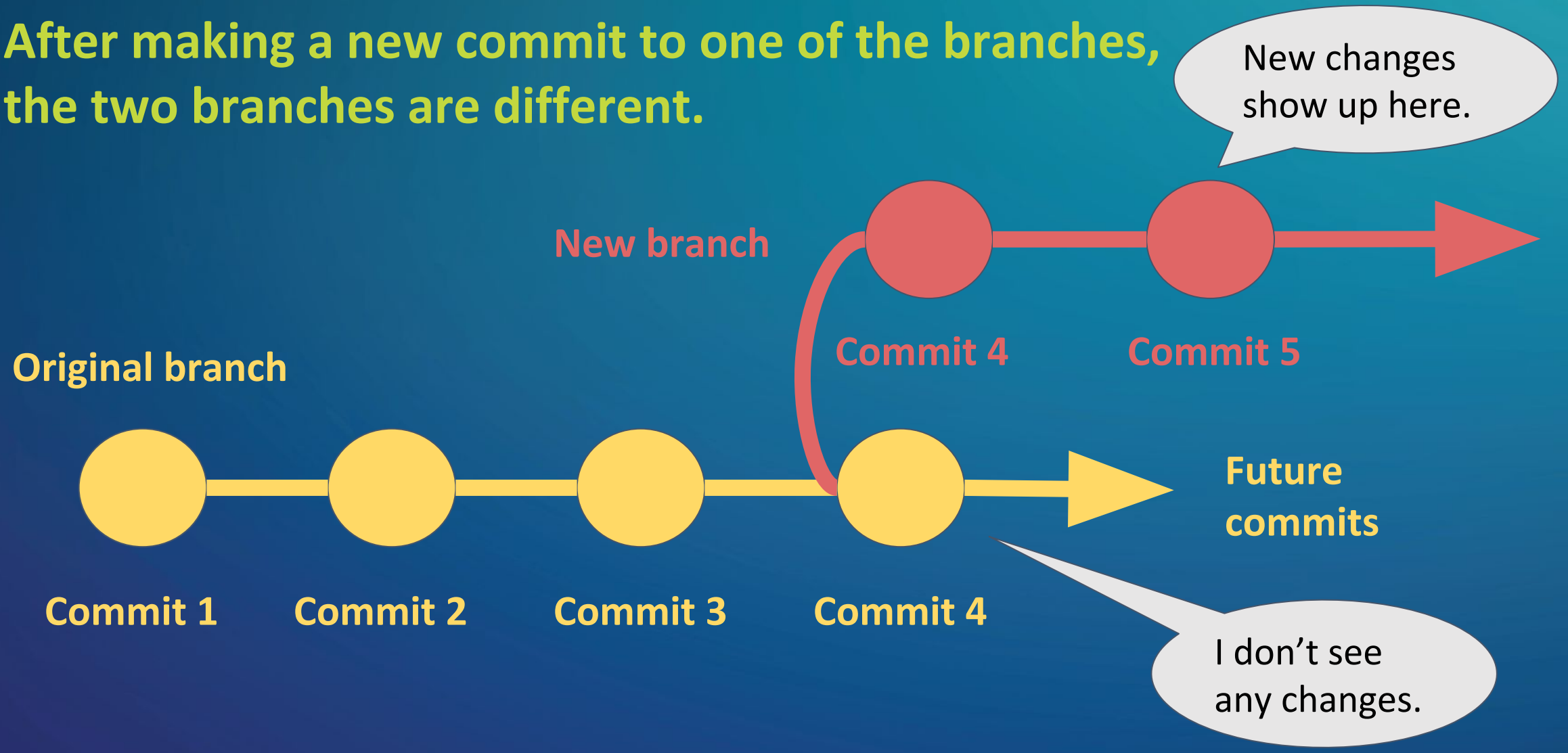
Step 3.2: Modifying the branch

Changes made in the new branch won't be reflected in the old branch. Try it and see!



Step 3.3: Committing branch changes

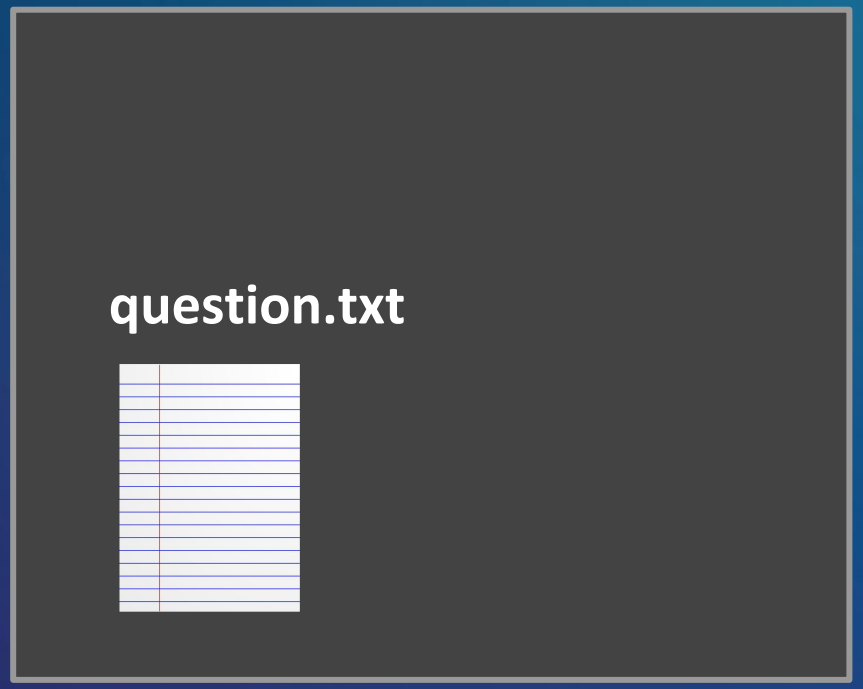
After making a new commit to one of the branches, the two branches are different.



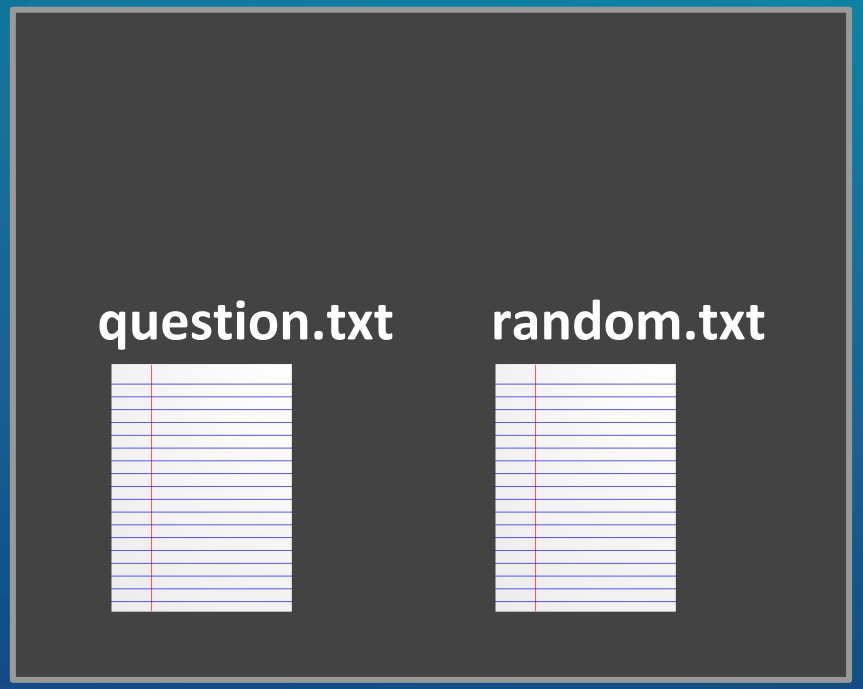
Step 3.4: Compare branches!

If you toggle between your two branches, you will see the files in the directory change!

Repo: Original branch

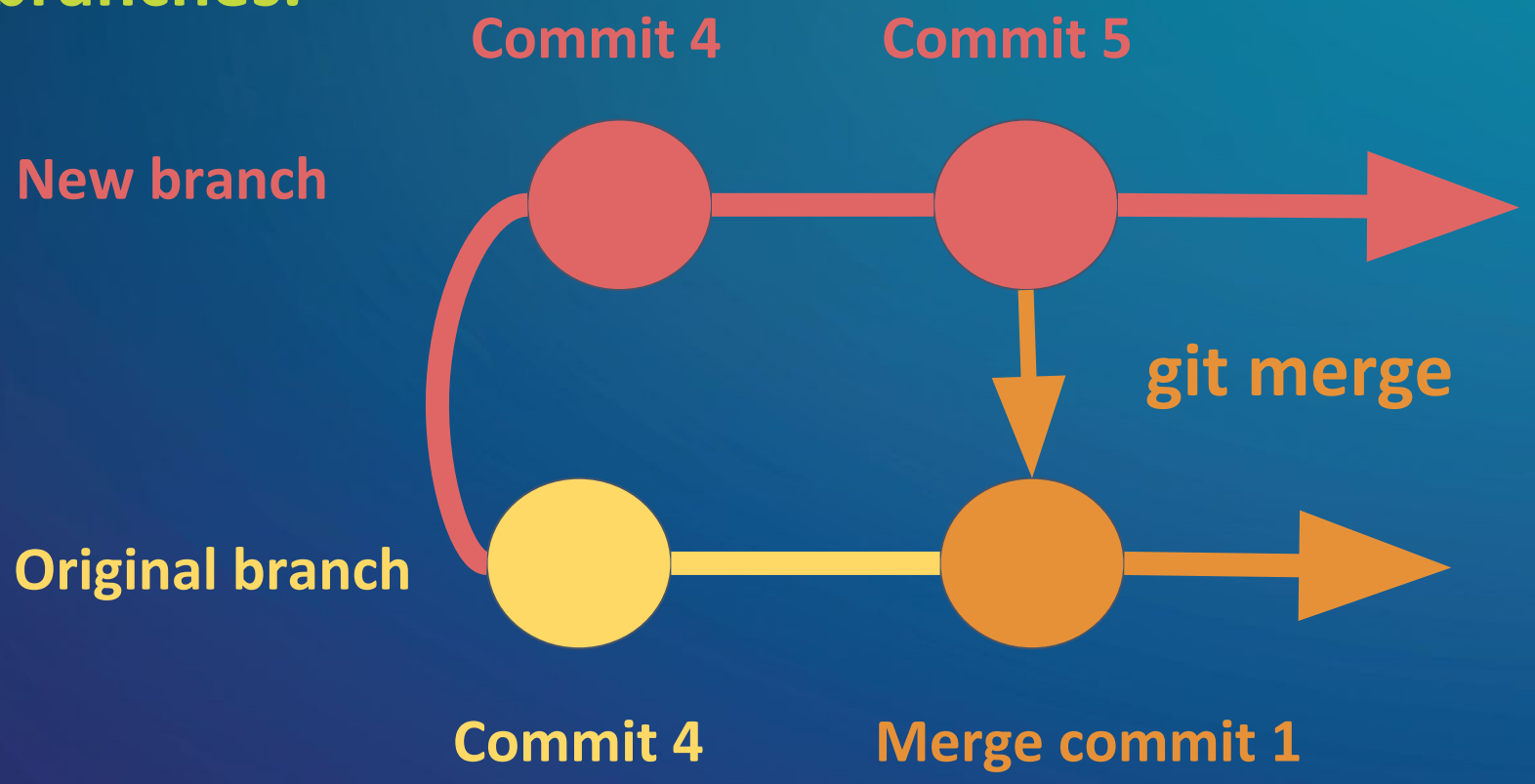


Repo: New branch



Step 3.5: Merge new into old

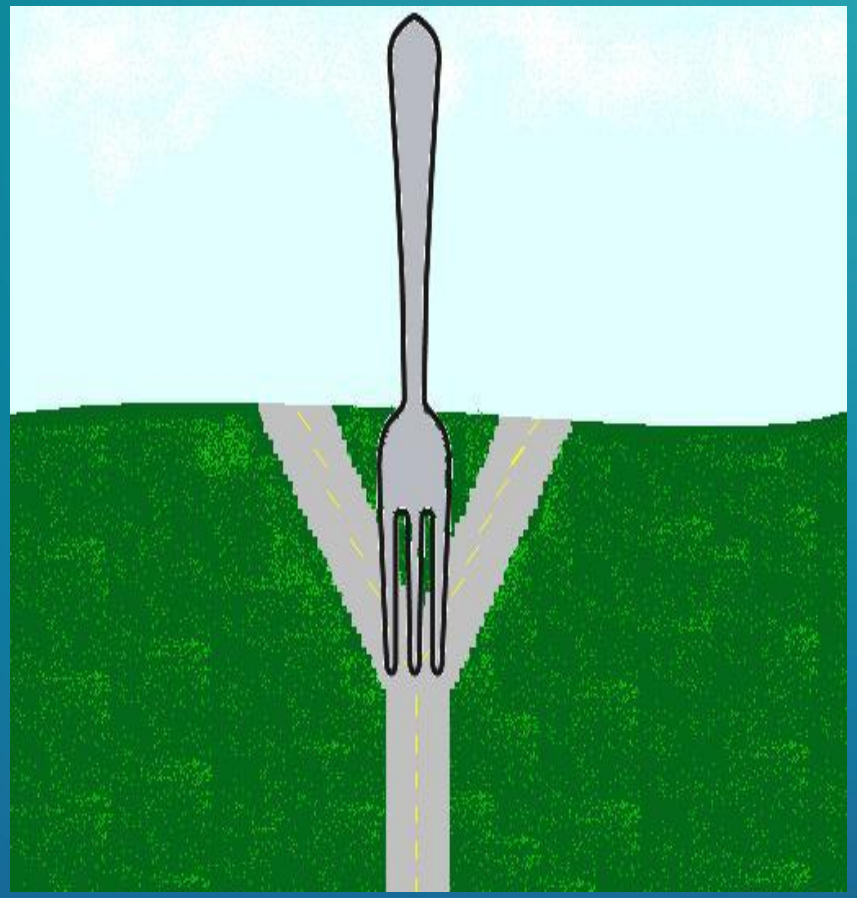
Merging pulls the changes from one branch into another. The merge commit reflects the state of both branches.



Step 3.6: Push changes to remote

Push changes to remote to see the merge reflected on GitHub.

Collaboration using GitHub



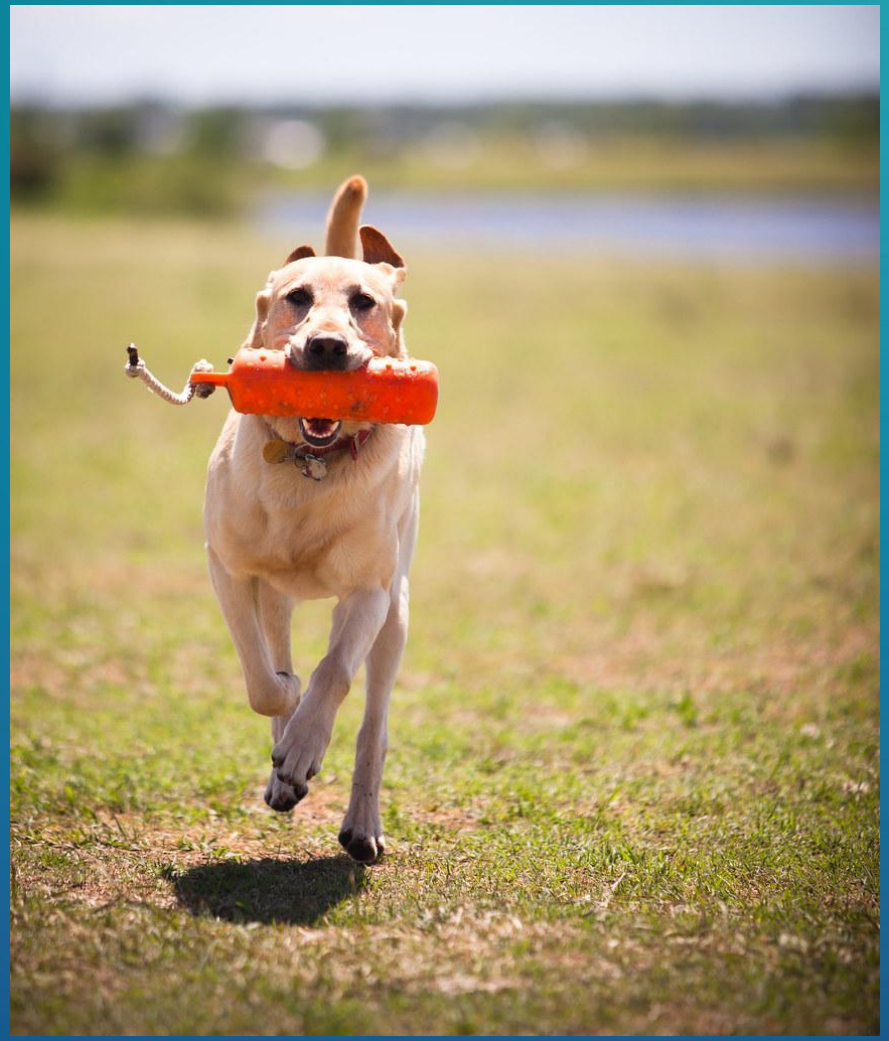
#GHC19

Fetch

`git fetch` checks if there are any commits available on the remote

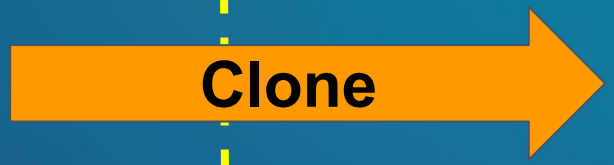
Rebase

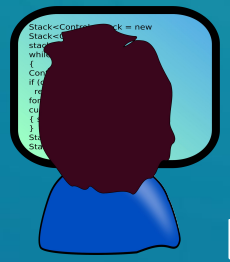
`git rebase` applies commits available on the remote to your local version



Always REBASE after FETCH

Fetch

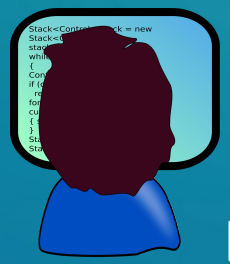




Local Version

t_1: commit id : “COMMIT-1”

Fetch



Local Version

New Code

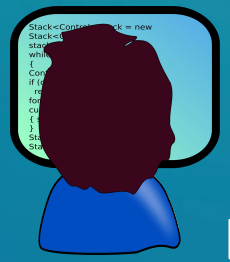
t_2: new commit added: “COMMIT-2”

t_1: commit id : “COMMIT-1”

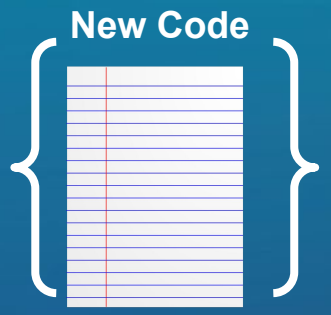
t_3: new commit “COMMIT-3”

#GHC19

Fetch



Local Version



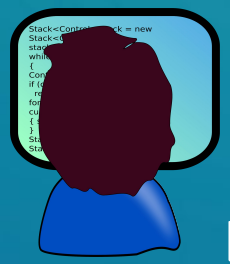
t_2: new commit added: "COMMIT-2"

t_1: commit id : "COMMIT-1"

t_3: new commit "COMMIT-3"

#GHC19

Fetch



Local Version

REBASE applies COMMIT-3 on top of COMMIT-2

“COMMIT-3”

“COMMIT-2”

t_2: new commit added: “COMMIT-2”

t_1: commit id : “COMMIT-1”

t_3: new commit “COMMIT-3”

Pull

"`git pull`" fetches ("`git fetch`") the new commits and merges ("`git merge`") them

Pull = Fetch + Merge

Upstream

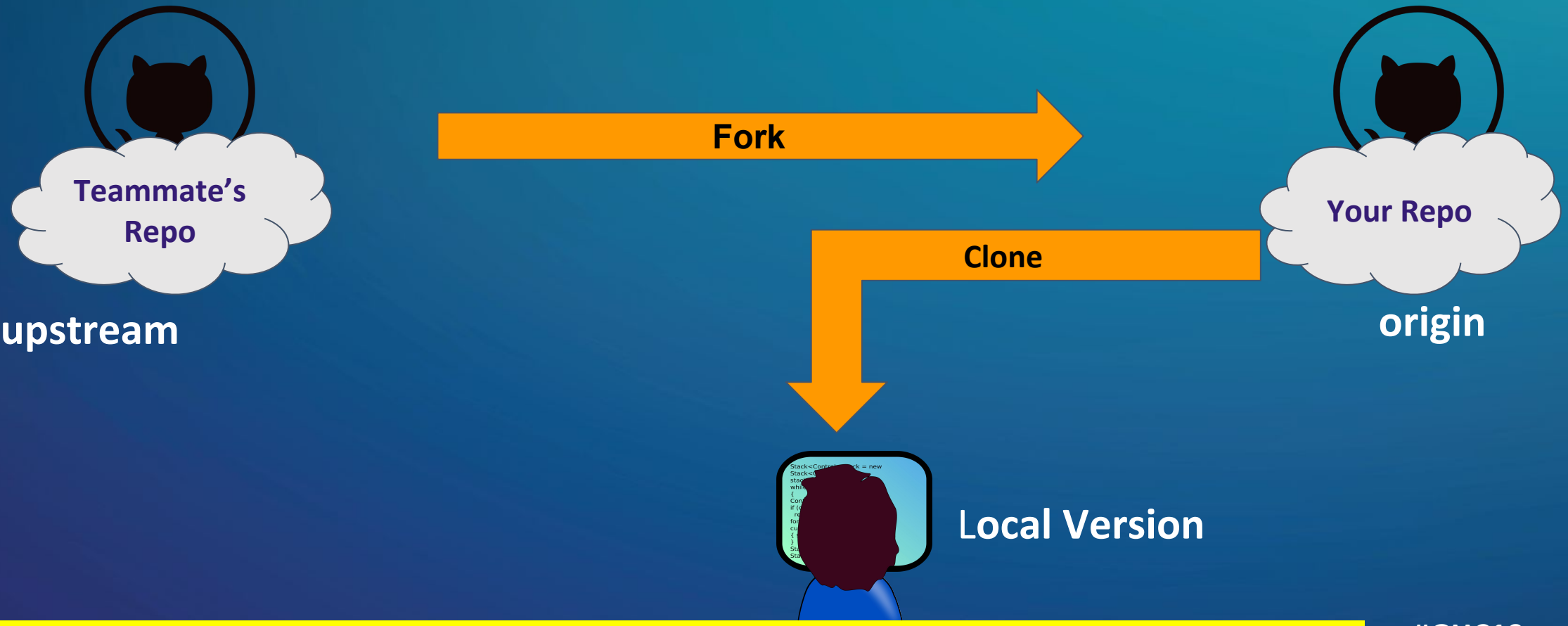
Original authors or maintainers of software that is distributed as source code

upstream

Fork

What? A fork is a copy of a repository

Why? Used for proposing changes



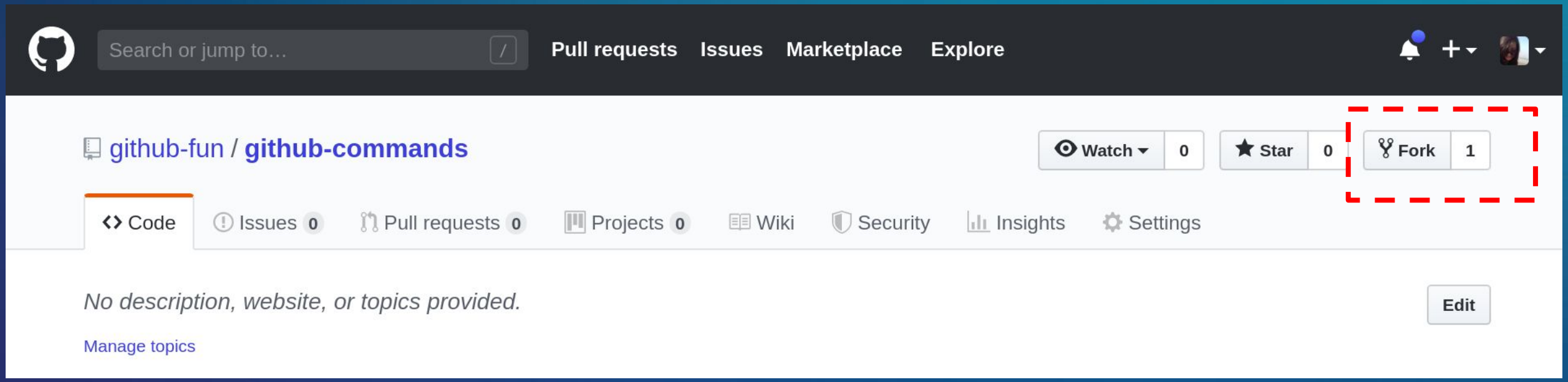
Follow along: https://github.com/github-fun/commands/blob/master/4-FORK.md

#GHC19

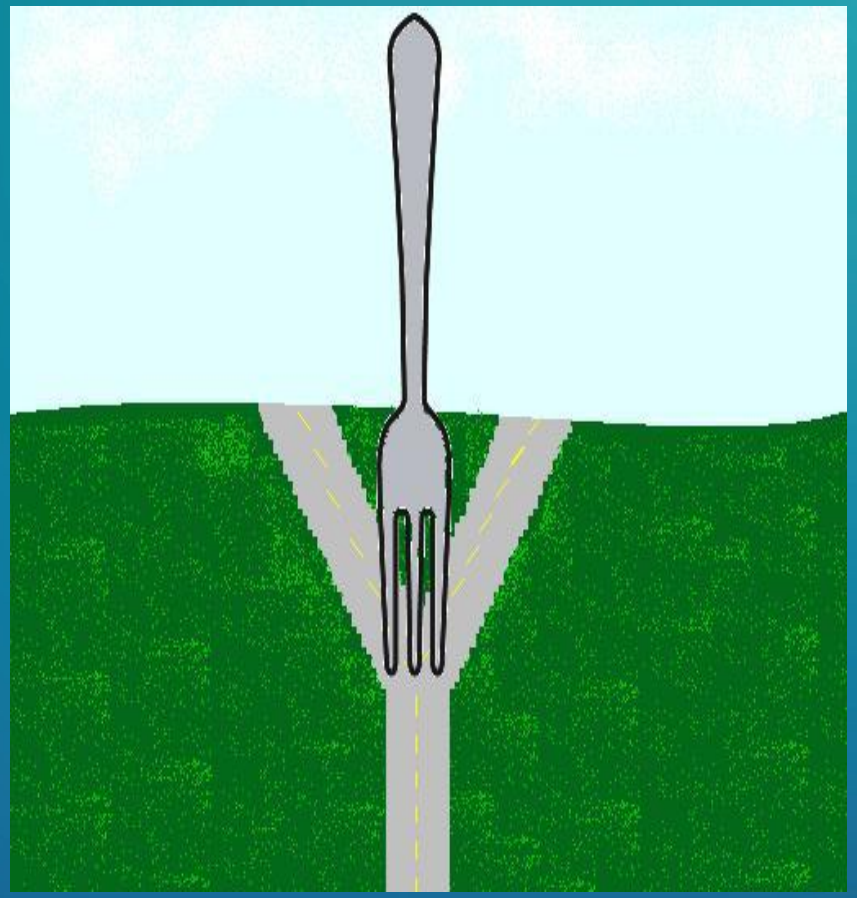
Forking - DIY

Follow steps in

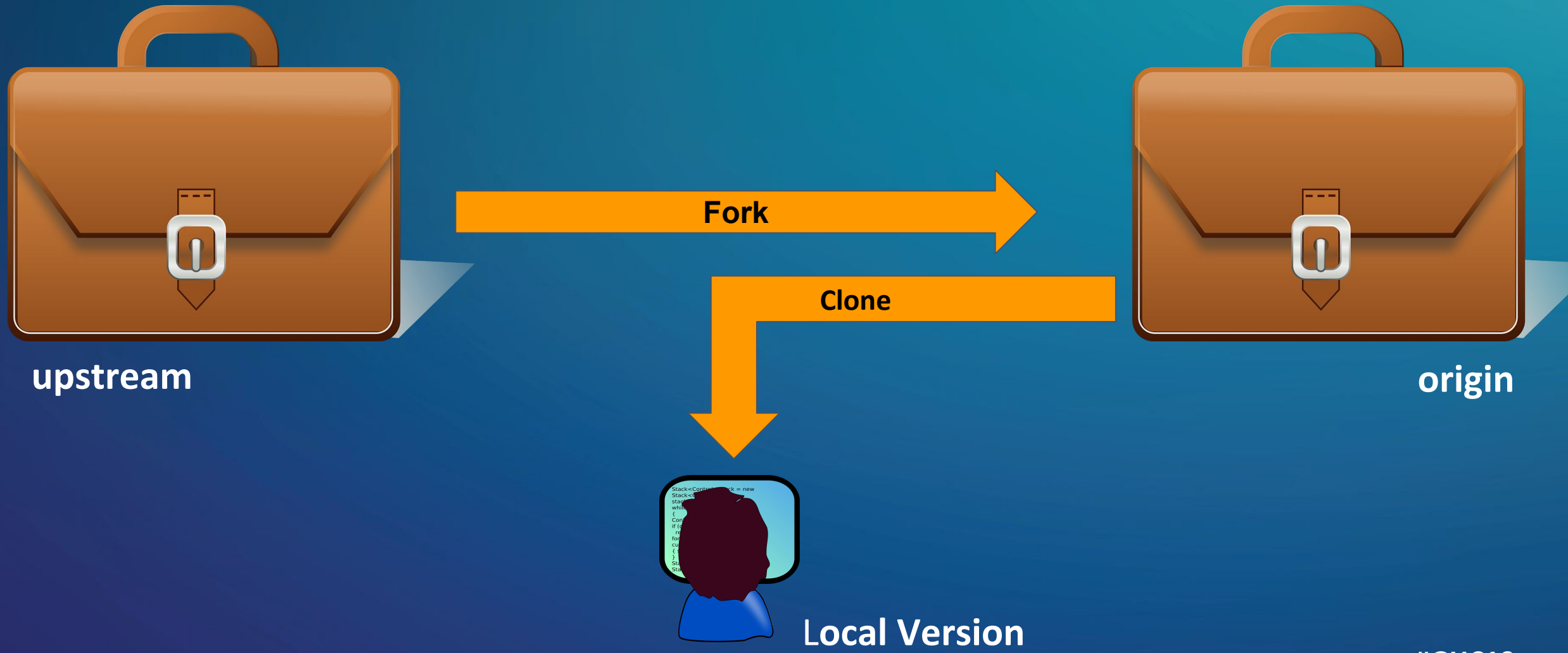
https://github.com/github-fun/commands/blob/master/4-FORK.md



Creating Pull Requests



Pull Request (PR) - Fork & Clone



Follow along: https://github.com/github-fun/commands/blob/master/5-PR.md

#GHC19

Pull Request - Make changes in “origin”

upstream

origin

Local Version

Make changes in your “origin” repo

Follow along: https://github.com/github-fun/commands/blob/master/5-PR.md

#GHC19

Pull Pequest - Propose changes by creating PR

upstream

Make Pull Request

origin

Local Version

Follow along: https://github.com/github-fun/commands/blob/master/5-PR.md

#GHC19

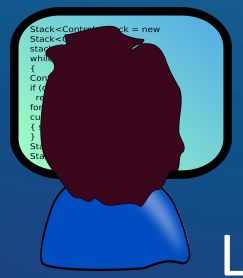
Pull Request - Review PR

upstream

Feedback

Pull Request

origin



Local Version

Follow along: https://github.com/github-fun/commands/blob/master/5-PR.md

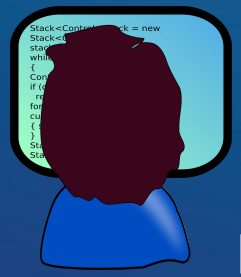
#GHC19

Pull request - Merge PR

Merge Pull Request

upstream

origin



Local Version

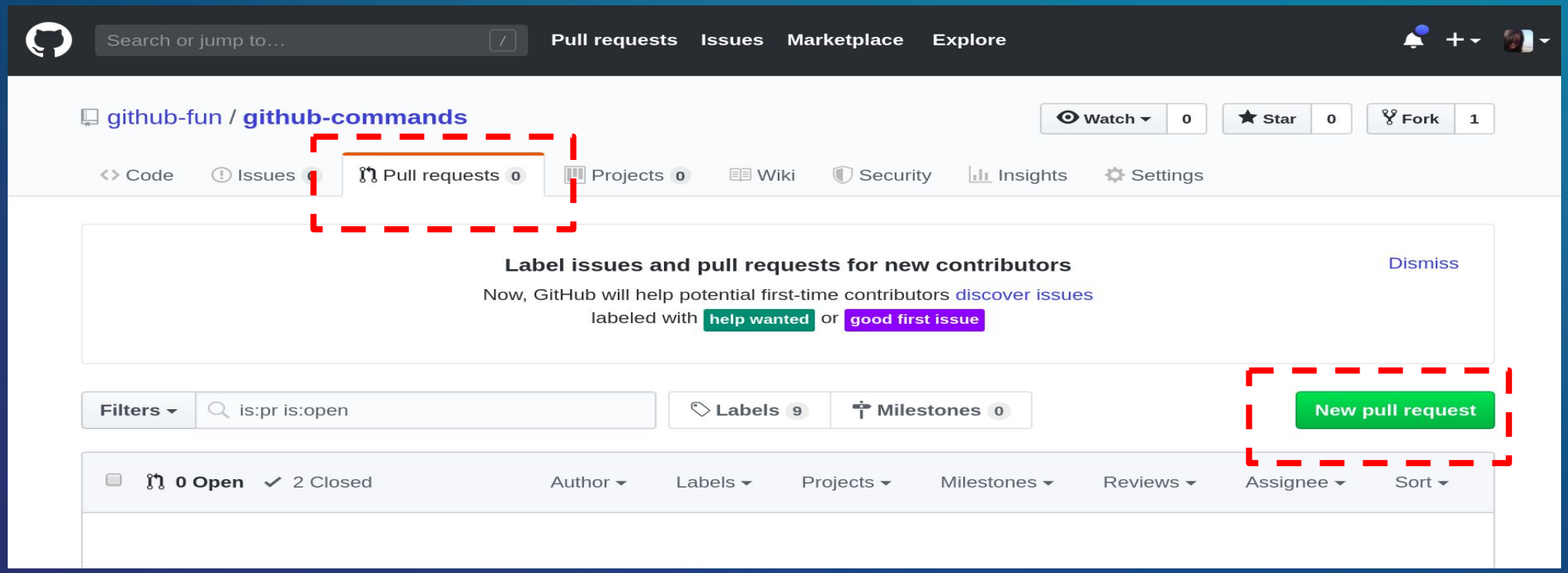
#GHC19

Follow along: https://github.com/github-fun/commands/blob/master/5-PR.md

Pull Request - DIY

Follow steps in

https://github.com/github-fun/commands/blob/master/5-PR.md



#GHC19

Ideas: How to get involved

Mozilla: https://codetribute.mozilla.org/

Pinax: https://github.com/pinax

Search issues by filter "first-timers-only"

Google Summer of Code : https://summerofcode.withgoogle.com/

Blog on beginner's friendly open source projects:
https://opensource.com/life/16/1/6-beginner-open-source

Appendix

GitHub repo https://github.com/github-fun

Tutorial for working on upstream

https://github.com/github-fun/github-commands/blob/master/FORK.md

Additional tips

https://github.com/github-fun/github-appendix/blob/master/README.md

Best git practices

https://github.com/github-fun/github-appendix/blob/master/Best_Practices.md

Questions

Thank You