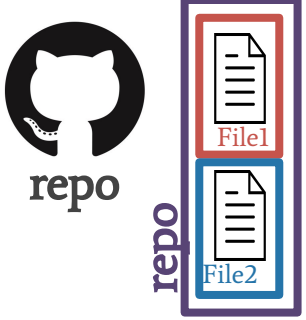
Plan:

1. Gain familiarity with basic version control processes
2. Introduce the basic version control verbs

Version Control: Verbs I

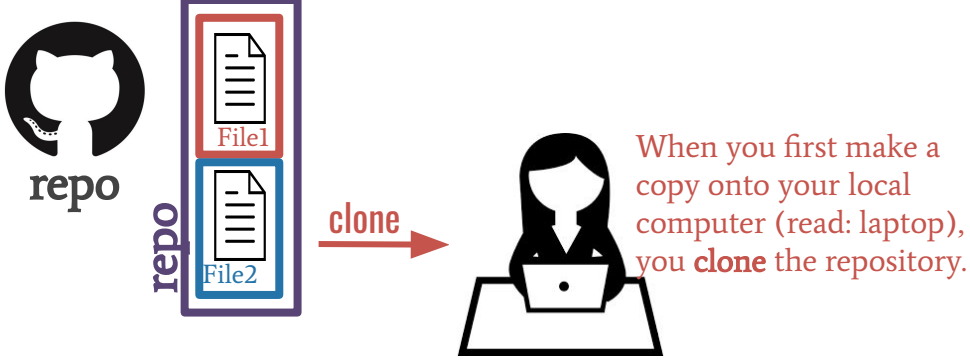
Shannon E. Ellis, Ph.D
UC San Diego

Department of Cognitive Science
sellis@ucsd.edu

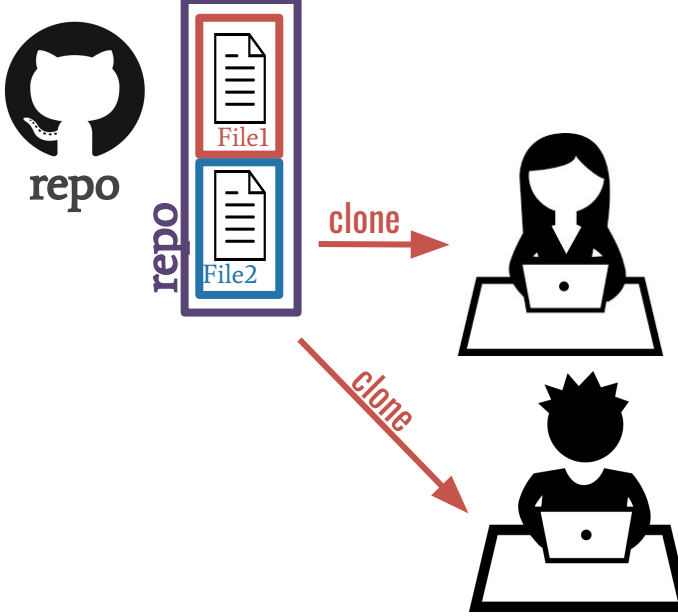


A **GitHub repo** contains all the files and folders for your project.

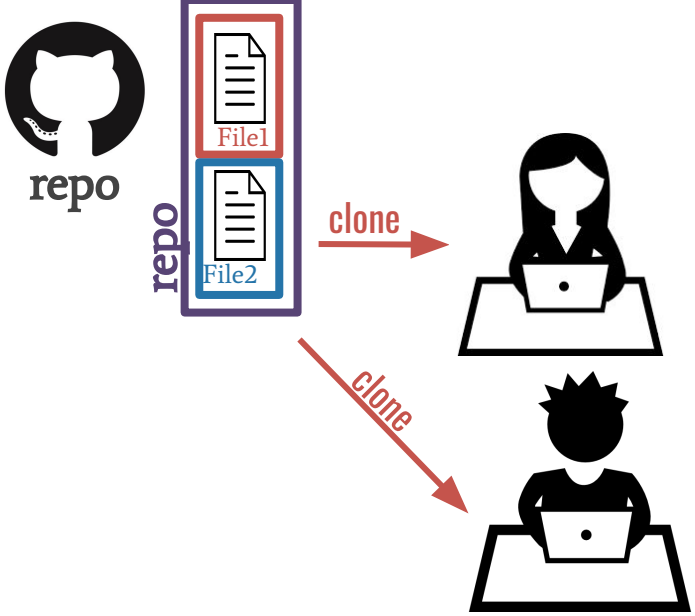
GitHub is a **remote host**. The files are geographically distant from any files on your computer.



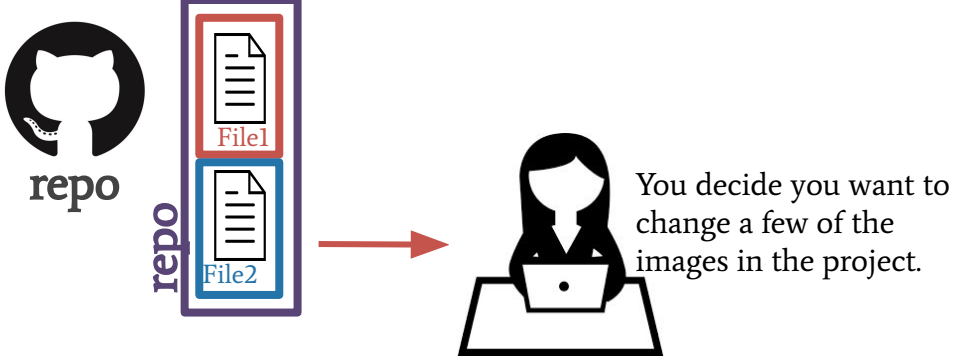
When you first make a copy onto your local computer (read: laptop), you **clone** the repository.



If someone else on your project cloned the repo at the same time, you would have identical copies of the project on each of your computers.

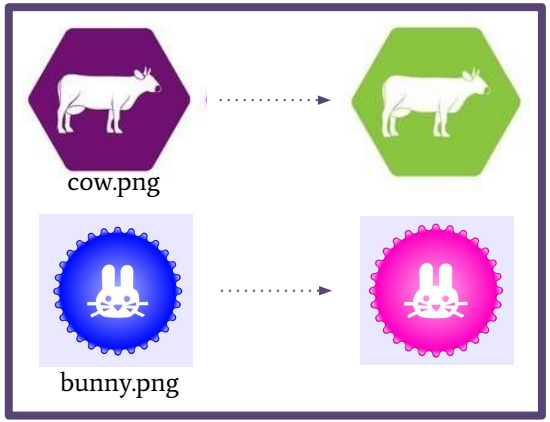


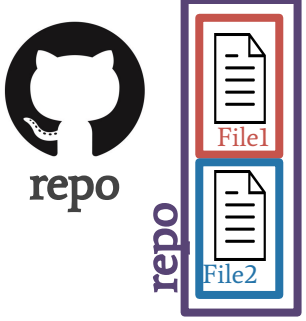
Yay! Everyone can work on the project!

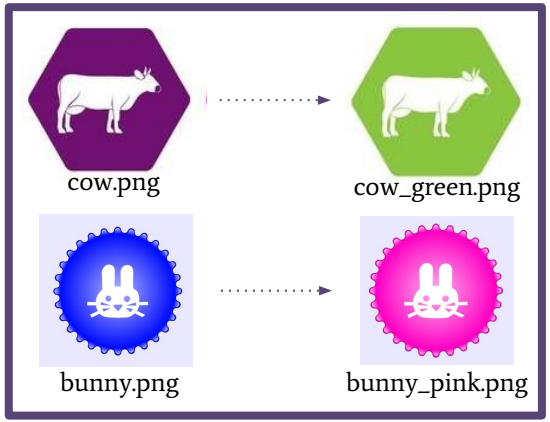


You decide you want to change a few of the images in the project.

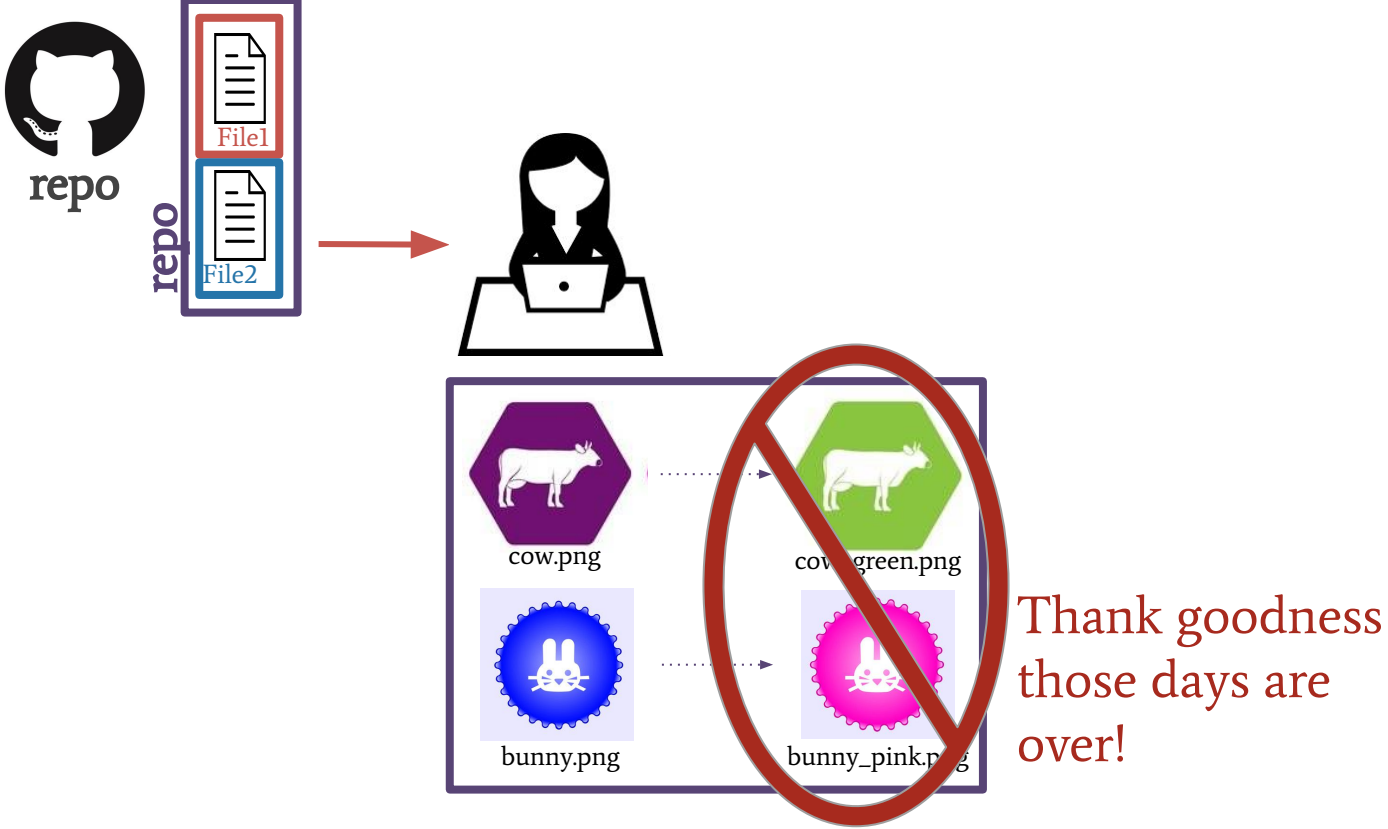
You decide you want to change a few of the images in the project.



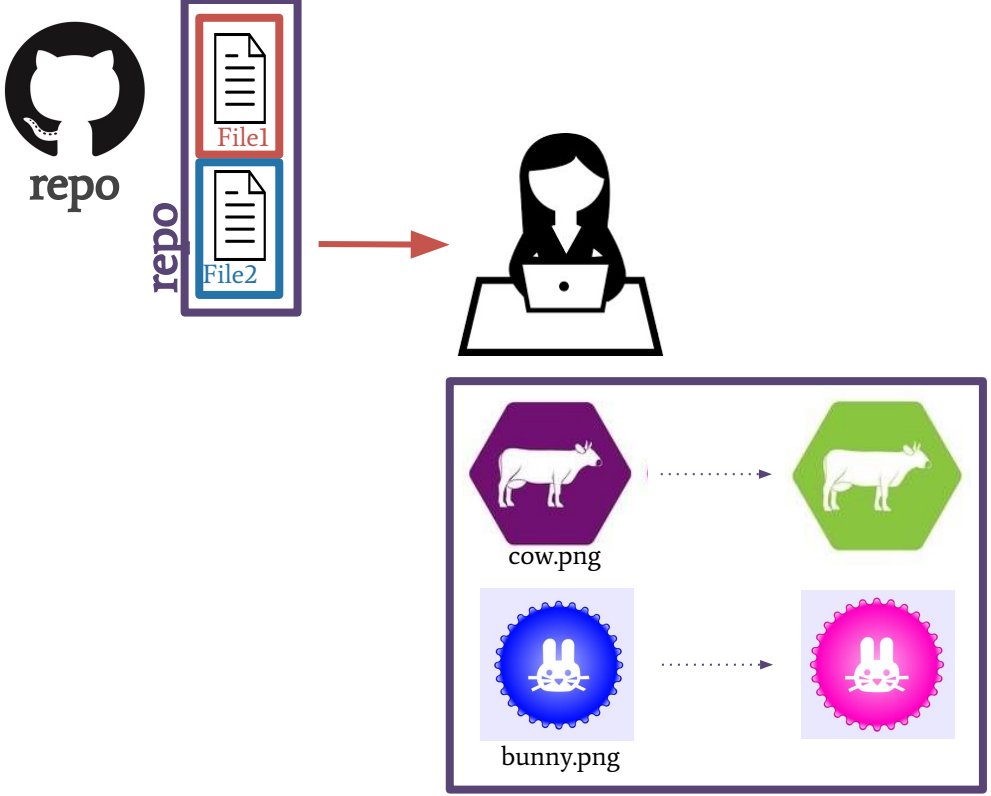




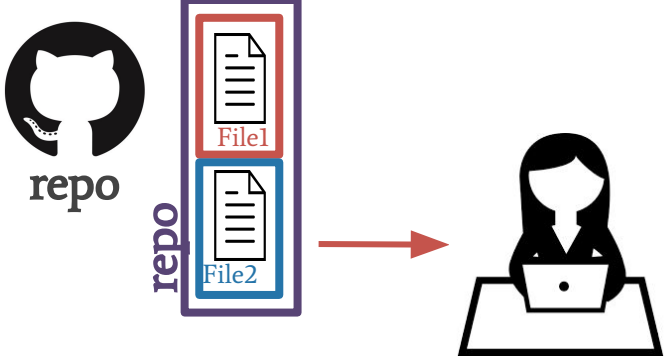
without git...you'd likely rename these files....



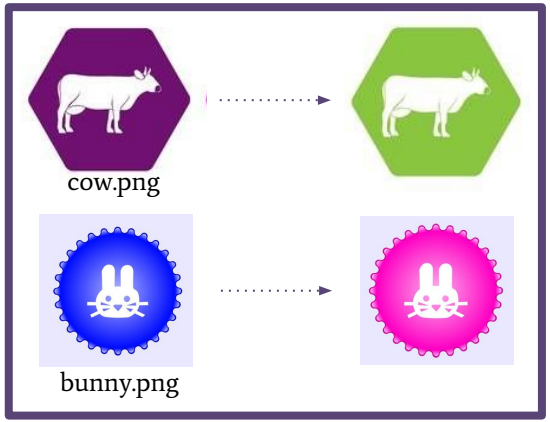
Thank goodness those days are over!



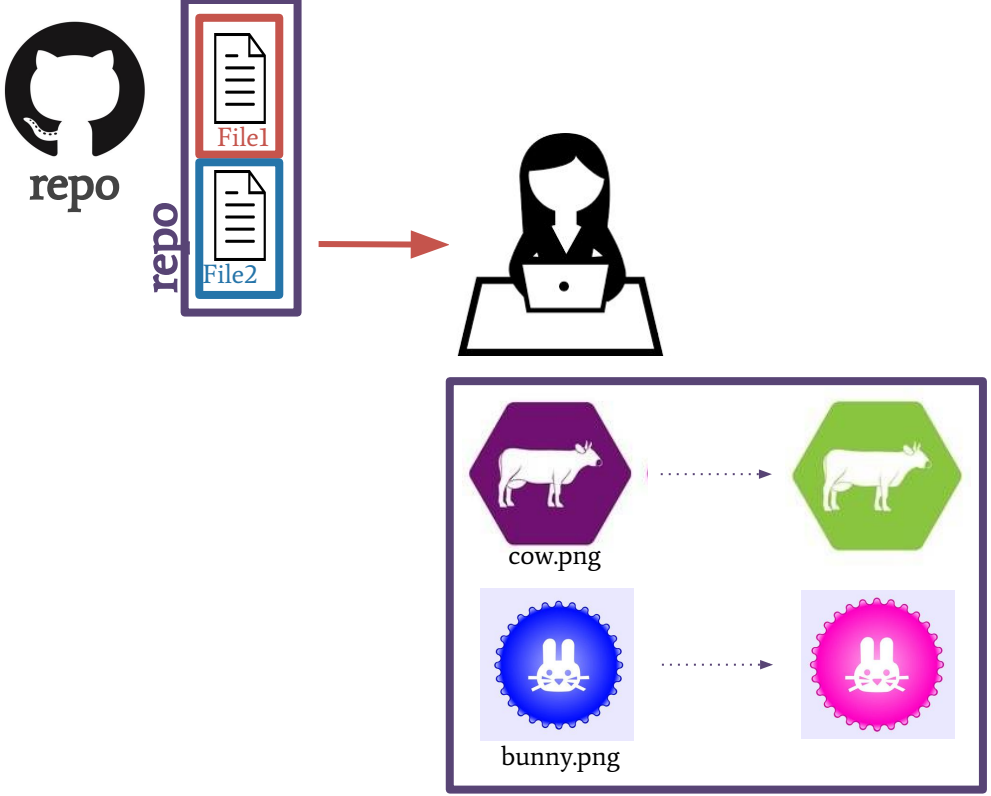
Instead, you tell git which files you'd like to keep track of using **add**. This process is called *staging*.



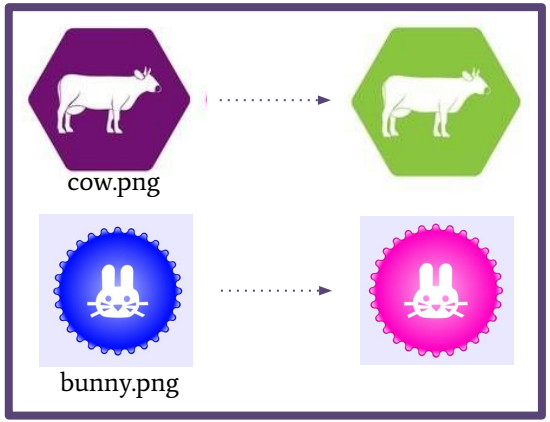
`git add file`	stages specified file (or folder)
`git add .`	stages **new and modified** files
`git add -u`	stages **modified and deleted** files
`git add -A`	stages **new, modified, and deleted** files
`git add *.csv`	Stages any files with .csv extension
`git add *`	Use with caution: stages everything



Instead, you tell git which files you'd like to keep track of using **add**. This process is called *staging*.

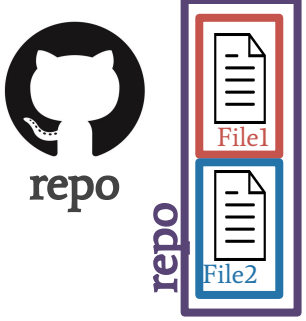


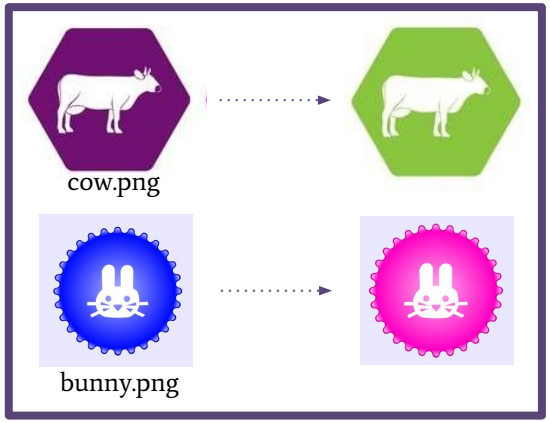
Then, you create a snapshot of your files at this point. This snapshot is called a **commit**.



Then, you create a snapshot of your files at this point. This snapshot is called a **commit**.

A **commit** tracks who, what, and when



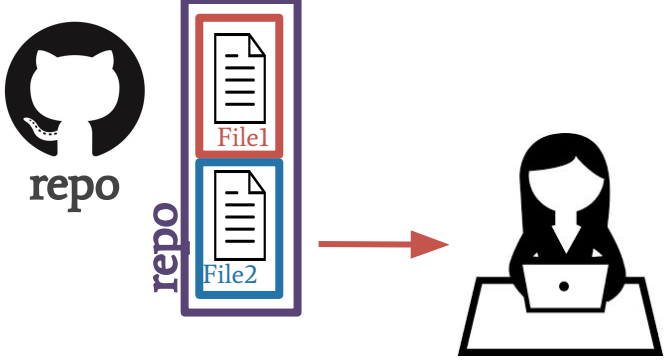


You can make commits more informative by adding a **commit message**.

Example: `git commit -m "changed colors for animal icons"`

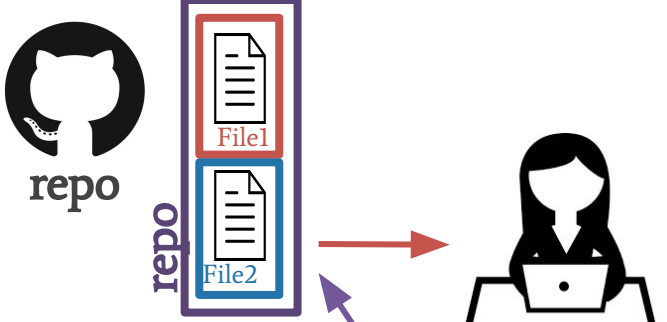
Then, you create a snapshot of your files at this point. This snapshot is called a **commit**.

A **commit** tracks who, what, and when



Shannon Ellis
11/28/18 3:28pm

changed colors for animal icons

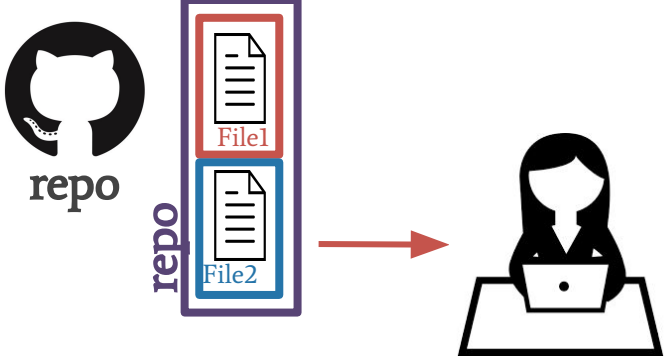


push

Remember, you're not the only one working on this project though! You want your teammates to have access to these changes! You **push** these changes back to the remote.

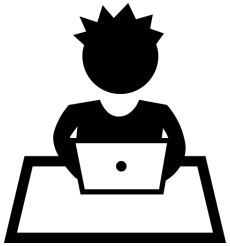
Shannon Ellis
11/28/18 3:28pm

changed colors for animal icons



Shannon Ellis
11/28/18 3:28pm

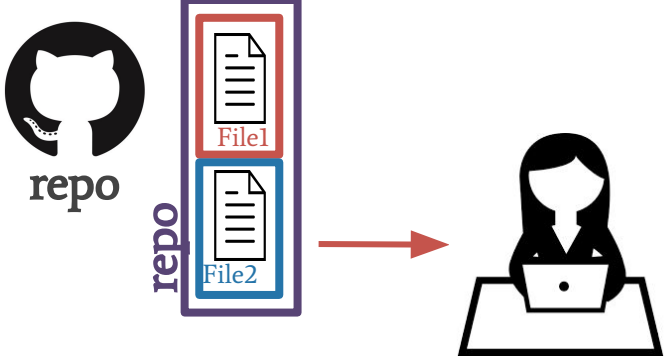
changed colors for animal icons



Your teammate is still working with the (out-of-date) copy he cloned earlier!

Shannon Ellis
11/28/18 3:28pm

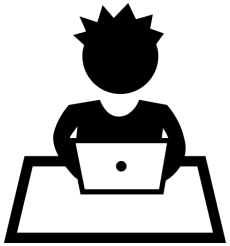
changed colors for animal icons



To catch up, your teammate will have to **pull** the changes from GitHub (remote)

Shannon Ellis
11/28/18 3:28pm

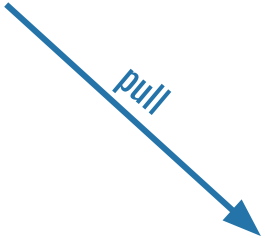
changed colors for animal icons



Your teammate is still working with the (out-of-date) copy he cloned earlier!

Shannon Ellis
11/28/18 3:28pm

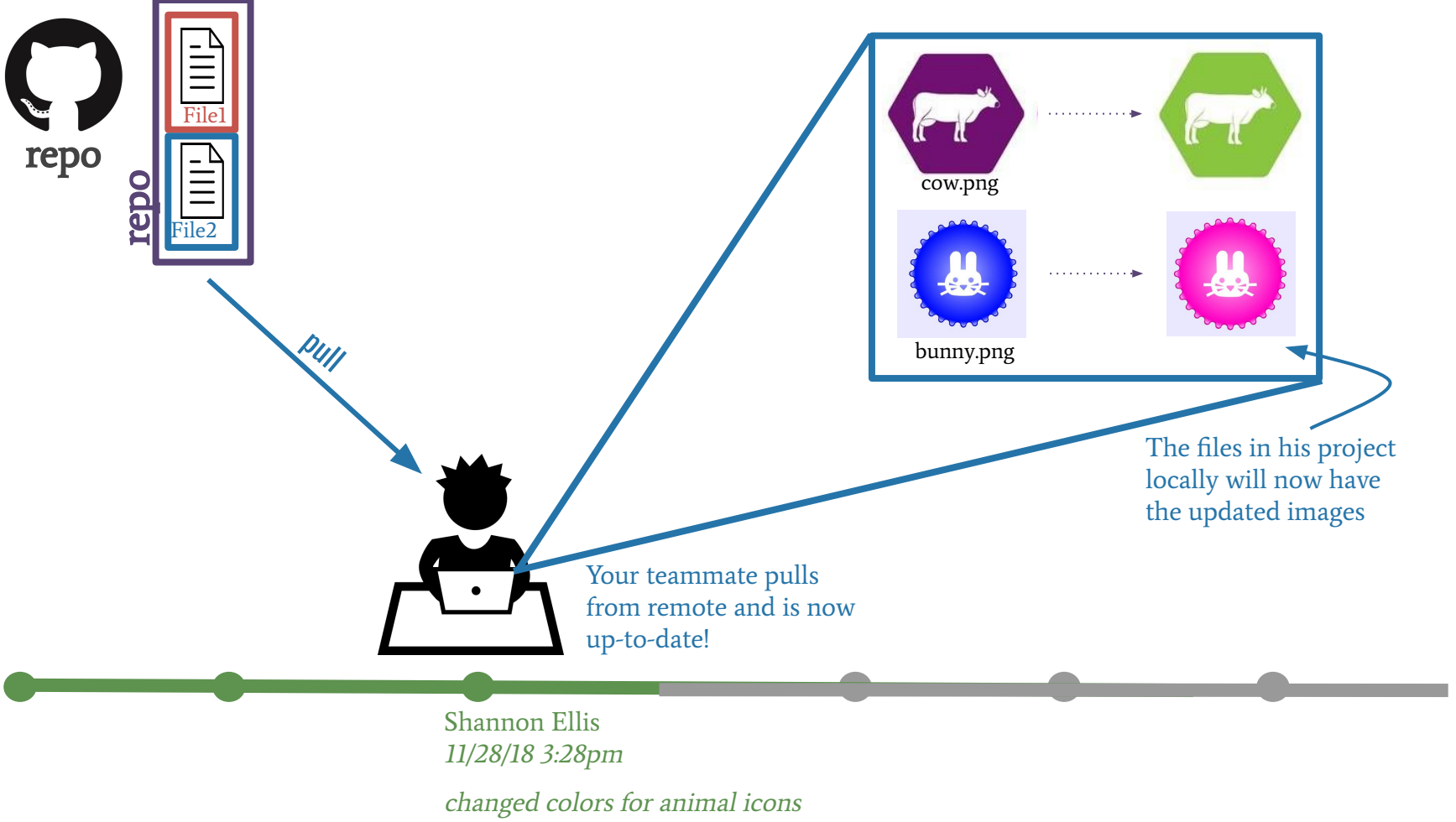
changed colors for animal icons

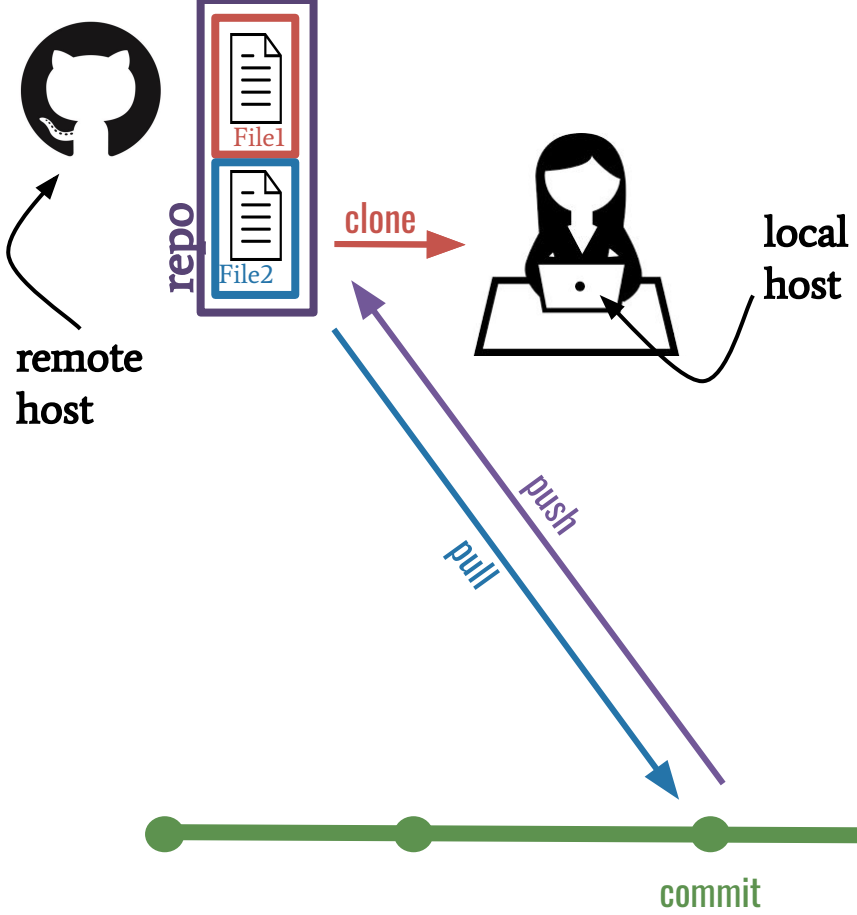


Your teammate pulls from remote and is now up-to-date!

Shannon Ellis
11/28/18 3:28pm

changed colors for animal icons


repo
repo
File1
File2
pull
cow.png
bunny.png
The files in his project locally will now have the updated images
Your teammate pulls from remote and is now up-to-date!
Shannon Ellis
11/28/18 3:28pm
changed colors for animal icons



Let's recap real quick!

repo - set of files and folders for a project
remote - where the repo lives
clone - get the repo from the remote for the first time
add - specify which files you want to stage (add to repo)
commit - snapshot of your files at a point in time
pull - get new commits to the repo from the remote
push - send your new commits to the remote