Introduction to Git and Github

Instructor: Aymeric Flaisler

Learning's objectives:

- Explain the basic function and purpose of version control
- Learn the difference between local and remote repository
- Become familiar with the basic/fundamental git commands and what each is used for:
  - fork
  - clone
  - add, commit
  - log
  - push, pull

Version Control:

- Using some sort of version control system is **nearly universal in the tech and data science world**, and Git is one of the most common.

Git:

- Is the **most popular** Version Control System
- It allows you to **share your personal projects** as well as **collaborate** on projects with others.

What's a version control system?

- A version control system is a repository of files with **monitored access**
  - Files are primarily **source code**, but can be of other types
- Version control is useful primarily because **every change to a repository is tracked**, along with who made it, why they made it, and references to any problems fixed or enhancements made by the change.
  - Provides the ability to **track changes over time**, and the ability to **reverse** any of them if necessary
  - Allows for **easy collaboration across teams**

Why use Git?

- Distributed (allows more freedom to work locally)
- Free, open source
- Collection of hosting services for Git repositories (Bit Bucket, Github)
- Extremely reliable

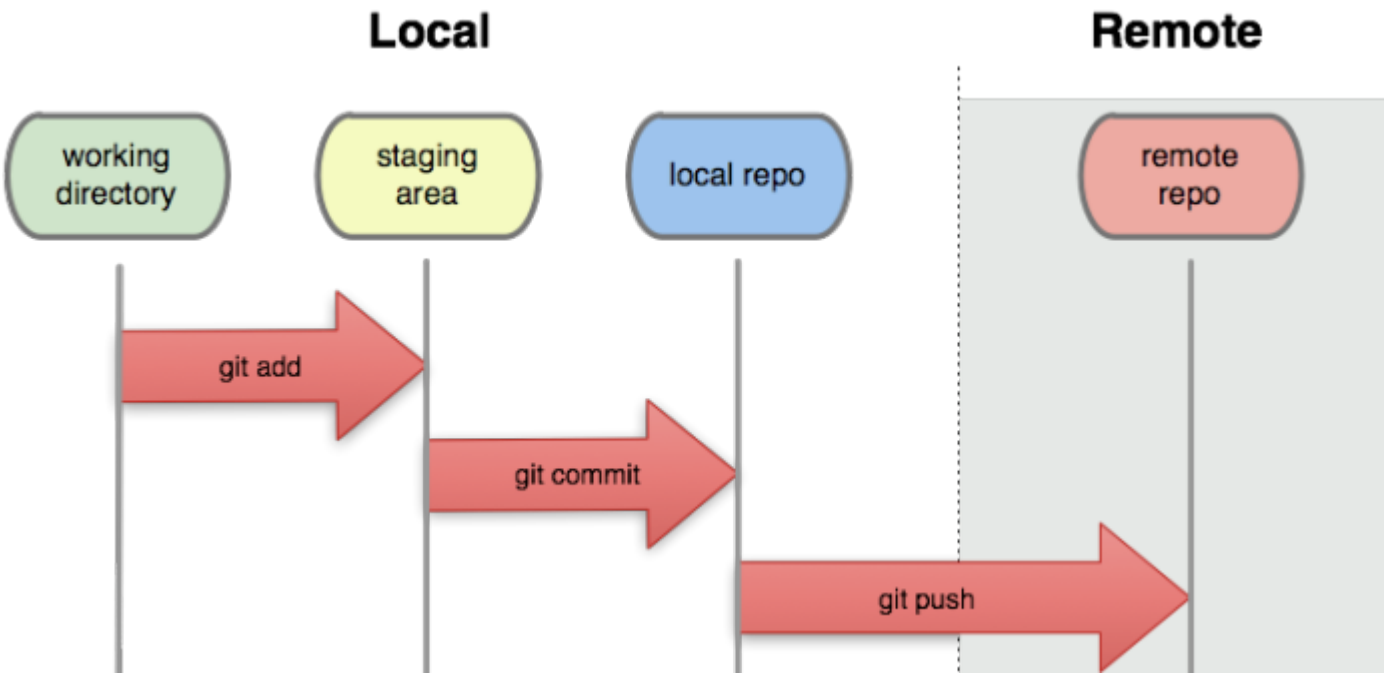
What's the difference between Git and Github?

- **Git is a version control system** (VCS), while Github is a **web-based hosting service for Git repositories**.
- Github is a web application built on the top of Git technology.
- Git exists independently of Github, while **the opposite is not true**.

Major Git commands include:

- **clone**: Makes a copy (clone) of a repository into a newly created directory, with a reference still pointing to the original repository.
- **add**: Add one or more files to the index (e.g. tell Git to keep track of these files)
- **commit**: Commit your changes, creating a “checkpoint” that can then be referenced or reverted back to later
- **status**: Displays paths that have differences between the index file and the current HEAD commit (last commit)
- **push**: Updates a remote copy of the repository with local changes
- **pull**: Updates a local copy of the repository with remote changes

Let's visualize how Git and Github interact:



- The command **add** tells your local repo to start tracking files.

```
git add week01/lesson-1.py # adding the lesson-1 in the folder
                           # week-1 to the index
```

- There are three important shortcuts:

```
git add . # adds all those paths to the staged changes if
          # they are either changed or are new and not ignored
```

```
git add -u # looks at all the already tracked files and stages
           # the changes to those files
```

When using the shortcut "." or "-a" TRIPLE check that you are not adding any sensitive information to your repo

- General best practice is to **commit early and often**. If you'd like to commit, but don't have a large enough piece of work to commit, then you can commit what you have and amend to the commit later.

  ```
  git commit -m "Add half of my function" # Half commit
  ```

- Now that you have saved your work locally it is **time to push** it to Github servers:

  ```
  git push
  ```

Let's create a repository (codealong):

- Go to https://git.generalassemb.ly (https://git.generalassemb.ly) and signup (if it is not already done)
- Send your username to Christoph so that he can grant you access to the course repositories
- Create a new repository (Click on the + in top right corner) and give it a name (note that you can make public or private)
- Add a gitignore prefiled for python
- click on 'clone or download' and copy the link
- Open iterm2 and type 'cd' + enter (making sure we are in your home directory)

- Enter: git clone + the link you just copied
- Type 'ls' to see your repository and cd into it

Let's **create a file and push it to github**:

- Do: vim hello_world.md
- Insert some test (you need to type 'i' to be in insert mode in VIM
- Press: ':wq' (this quit vim insert mode and save the text)

Time to use git:

- Now add your file to your repo (git add)
- Save it locally: git commit -m "first commit"
- and push it: git push

To **avoid entering your credentials every times**: git config --global credential.helper cache

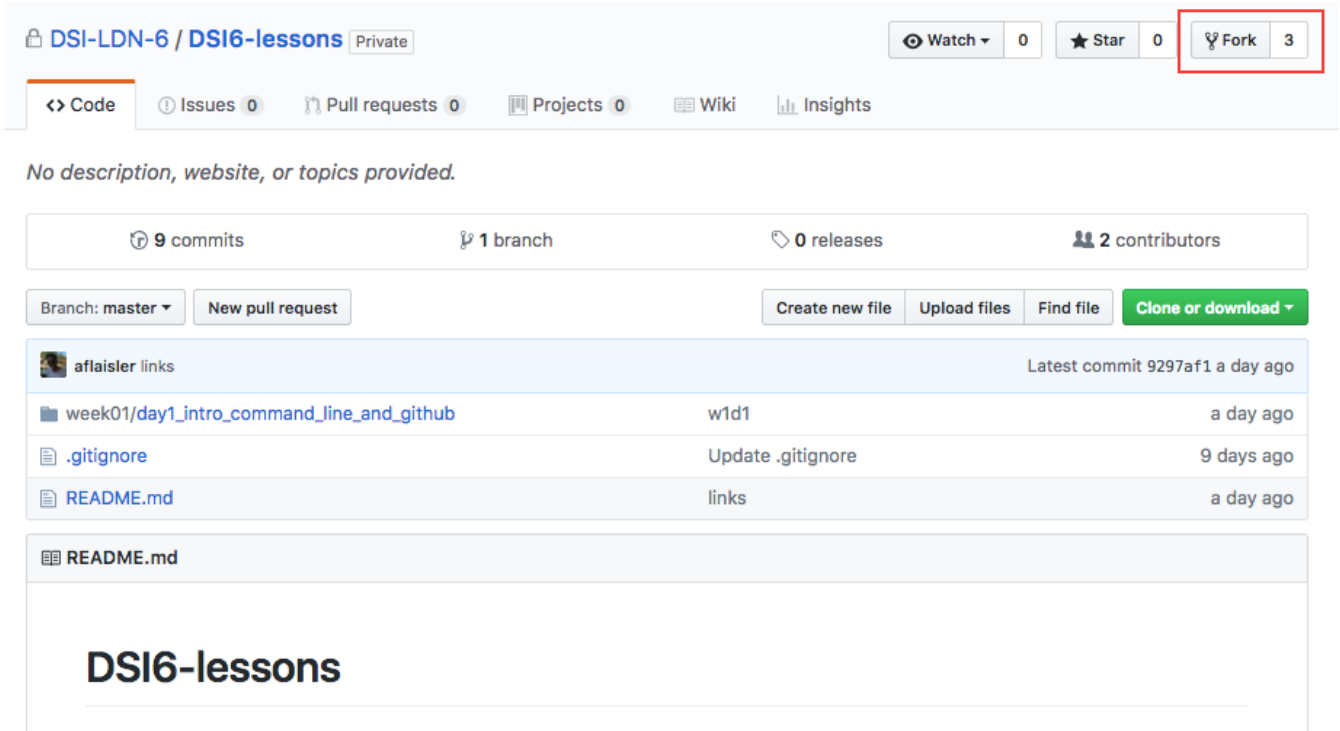
Github web app:

Two important commands are:

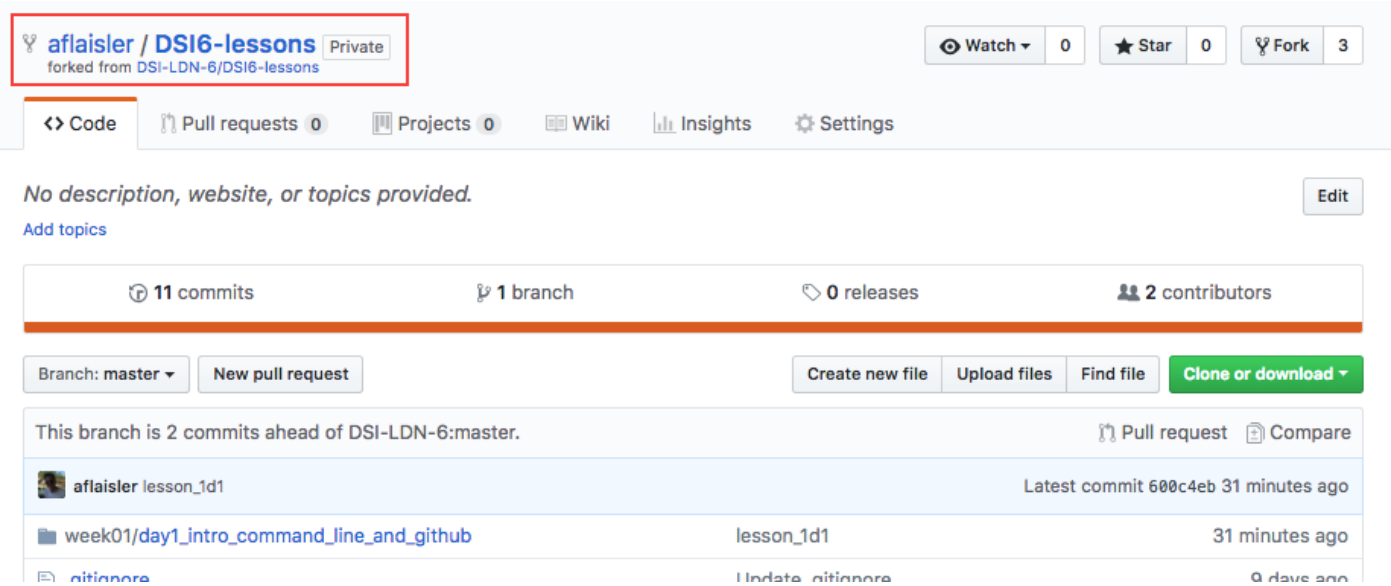
- **fork**: Makes a copy of a repository onto your personal Github account (it's like the Github version of cloning), but without a reference still pointing to the original repository.
- **pull request**: Issued to try to update a repository with changes from another copy of the repository (will go through it Friday).

Let's fork the lessons reporsitory into your github account:

- Go to this page: [https://git.generalassemb.ly/DSI-LDN-6/DSI6-lessons (https://git.generalassemb.ly/DSI-LDN-6/DSI6-lessons)](https://git.generalassemb.ly/DSI-LDN-6/DSI6-lessons)
- Click on fork:



Now that we've forked the repository into you account, let's clone it locally (Note: We're now looking at our personal copy of the repository):



- So, we issued a fork in our browser to get **our own personal copy** of the repository
- Then we need to clone it in our terminal to get that personal copy on **our local machine**: `git clone <repository url>`

Note 1: This is the directory where all the lessons will be. Choose the location strategically (ie: not in a messy folder). I recommend to create a folder 'GA' somewhere in your computer where you will clone all the repositories from this course.

Note 2: Try to avoid as much as possible having a git repository within another one. The right way to do it requires to use git submodules which is outside the scope of this lesson.

Git remote:

The command `git remote` allows you to manage the set of repositories you are working with.

By default, when you clone a repository, `git` add the **origin of the url**. If you type 'git remote' in the lesson repository it should returns 'origin'.

If you do 'git remote get-url origin' it will return the url of your **remote repository**.

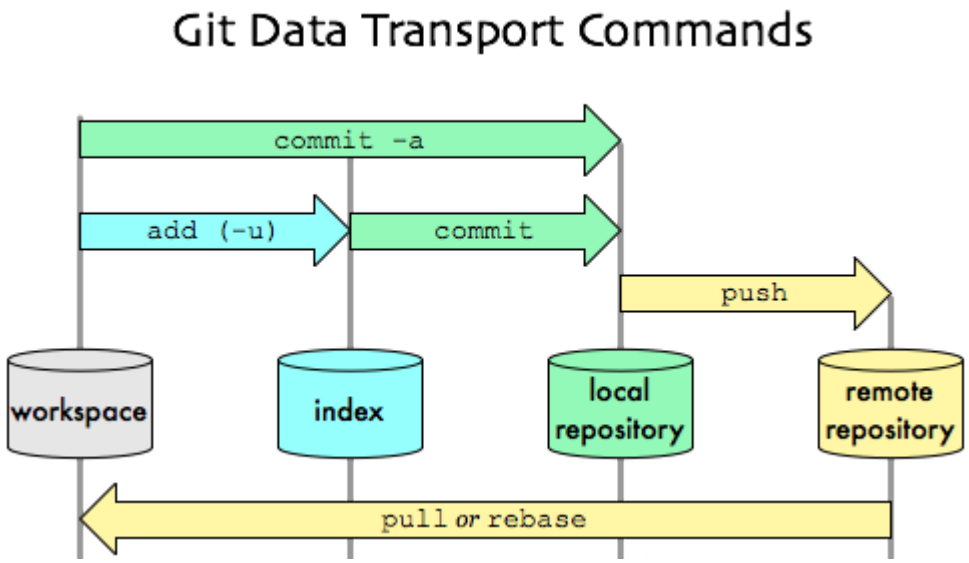
Let's add the course repository url:

- Enter: 'git remote add upstream https://git.generalassemb.ly/DSI-LDN-6/DSI6-lessons' (https://git.generalassemb.ly/DSI-LDN-6/DSI6-lessons')
- Confirm you correctly added it using 'git remote get-url upstream'

Breakout (5min): In pairs, discuss what we just did (git add/commit/push). Why version control are great tools? What's the difference between the upstream / origin / local repositories?

Git pull:

The " git pull " command will fetch and replay the changes from the **remote master branch** since it diverged from the local master (i.e., E) until its current commit:



In order to get the lessons, every morning, you will have to do (from the lessons repo):

- `git pull upstream master`
- `git push origin master`

Question (2min): Why one command refers to 'upstream master' and the other one to 'origin master'? In other words, what's the difference between `upstream` and `origin` ?

Exercise 1 (10 min):

1. Find the repo "DSI-LDN-6/project-submissions"
2. Fork it to your account
3. Clone it to your local machine
4. Create a folder called FirstName-LastName (ie: Richard-Smith)
5. In this folder create a markdown file named Hello-World.md with the text "This is my first pull request" in it
6. Push it to your account

Note: We will use this repository to evaluate your project works.

Exercise 2 (10 min):

1. Repeat the process with this repository: https://git.generalassemb.ly/DSI-LDN-5/resource-datasets (https://git.generalassemb.ly/DSI-LDN-5/resource-datasets)

Note: This repository contains most of the data set we are going to use throughout the course.

In [ ]:

```
1
```