

Introduction to Git and GitHub

General Assembly – Data Science

Agenda

- I. Introduction
- II. Exploring GitHub
- III. Using Git with GitHub
- IV. Contributing on GitHub
- V. Bonus Content

I. Introduction

Why learn version control?

- Version control is useful when you write code, and data scientists write code
- Enables teams to easily collaborate on the same codebase
- Enables you to contribute to open source projects
- Attractive skill for employment

What is Git?

- Version control system that allows you to track files and file changes in a repository (“repo”)
- Primarily used by software developers
- Most widely used version control system
 - Alternatives: Mercurial, Subversion, CVS
- Runs from the command line (usually)
- Can be used alone or in a team

What is GitHub?

- A website, not a version control system
- Allows you to put your Git repos online
 - Largest code host in the world
 - Alternative: Bitbucket
- Benefits of GitHub:
 - Backup of files
 - Visual interface for navigating repos
 - Makes repo collaboration easy
- “GitHub is just Dropbox for Git”
- Note: Git does not require GitHub

Git can be challenging to learn

- Designed (by programmers) for power and flexibility over simplicity
- Hard to know if what you did was right
- Hard to explore since most actions are “permanent” (in a sense) and can have serious consequences
- We’ll focus on the most important 10% of Git

II. Exploring GitHub

GitHub setup

- There's nothing to install
 - “GitHub for Windows” & “GitHub for Mac” are GUI clients (alternatives to command line)

Navigating a GitHub repo (1 of 2)

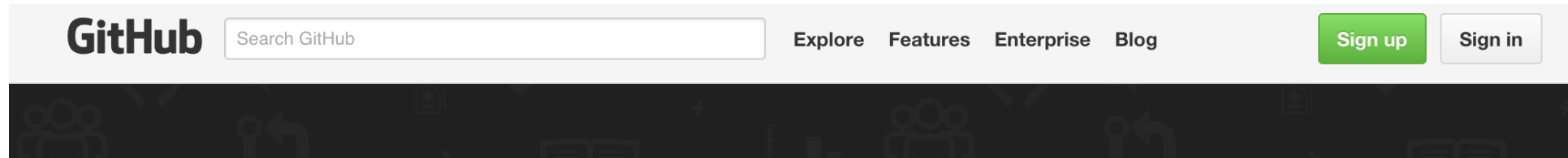
- Example repo: github.com/sinanuozdemir/sfdat22
- Account name, repo name, description
- Folder structure
- Viewing files:
 - Rendered view (with syntax highlighting)
 - Raw view
- README.md:
 - Describes a repo
 - Automatically displayed
 - Written in Markdown

Navigating a GitHub repo (2 of 2)

- Commits:
 - One or more changes to one or more files
 - Revision highlighting
 - Commit comments are required
 - Most recent commit comment shown by filename
- Profile page

Creating a profile

- Click on the signup button on the top-right
- Choose a plan (one of them is free)




- **Remember your email and password!!!! You will need it again soon!!!!**


III. Using Git with GitHub

Cloning a GitHub repo

- Cloning == copying to your local computer
 - Like copying your Dropbox files to a new machine
- First, change your working directory to where you want the repo you created to be stored: `cd`
- Then, clone the repo: `git clone <URL>`
 - Get HTTPS or SSH URL from GitHub (ends in .git)
 - Clones to a subdirectory of the working directory
 - No visual feedback when you type your password
- Navigate into the repo (`cd`) then list the files (`ls`)

The url is on the repo page

 This repository Search Pull requests Issues Gist


 **sinanuoazdemir** / **sfdat22** Unwatch 5 Star 4 Fork 2





[Code](#) [Issues 0](#) [Pull requests 0](#) [Wiki](#) [Pulse](#) [Graphs](#) [Settings](#)


SF DAT 22 Course Repository — Edit

6 commits1 branch0 releases1 contributor

Branch: master New pull request New file Upload files Find file HTTPS <https://github.com/sinanuoazdemir/sfdat22> Download ZIP

 **sinanuoazdemir** Adding material for first class Latest commit d74267e 2 days ago

 notebooks	Adding material for first class	2 days ago
 slides	Adding material for first class	2 days ago
 .gitignore	Ad	3 days ago
 README.md	Adding material for first class	2 days ago

 README.md

SF DAT 22 Course Repository

First Clone

- First we will clone the main class repo!
- You will need to do this to stay up to date with all of class info

Before Cloning

- Move into a Directory that you want to store the info for the next 10 weeks
- For my example, I will use my Desktop

Before Cloning -- use pwd and ls

```
[Sinans-MacBook-Pro:Desktop sinanozdemir$ pwd  
/Users/sinanozdemir/Desktop  
[Sinans-MacBook-Pro:Desktop sinanozdemir$ ls  
saved_texts.rtf  
Sinans-MacBook-Pro:Desktop sinanozdemir$
```

During Cloning

```
[Sinans-MacBook-Pro:Desktop sinanozdemir$ git clone https://github.com/sinanuoazdemir/sfdat22.git  
Cloning into 'sfdat22'...  
remote: Counting objects: 30, done.  
remote: Compressing objects: 100% (19/19), done.  
remote: Total 30 (delta 8), reused 30 (delta 8), pack-reused 0  
Unpacking objects: 100% (30/30), done.  
Checking connectivity... done.  
Sinans-MacBook-Pro:Desktop sinanozdemir$
```

git clone <https://github.com/sinanuoazdemir/sfdat22.git>





After Cloning

```
[Sinans-MacBook-Pro:Desktop sinanozdemir$ ls  
saved_texts.rtf sfdat22  
Sinans-MacBook-Pro:Desktop sinanozdemir$
```

- You have a new folder!

Same as on Github!

```
[Sinans-MacBook-Pro:Desktop sinanozdemir$ cd sfdat22/  
[Sinans-MacBook-Pro:sfdat22 sinanozdemir$ ls  
README.md      notebooks      slides  
Sinans-MacBook-Pro:sfdat22 sinanozdemir$
```

 notebooks	Adding material for first class	2 days ago
 slides	Adding material for first class	2 days ago
 .gitignore	Ad	3 days ago
 README.md	Adding material for first class	2 days ago

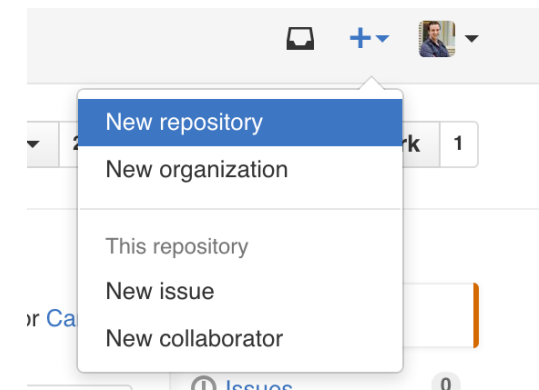
- **cd** into it and **ls**
- **Try this now! (take 5 minutes)**
- The **.gitignore** file is “ignored” and is only there to prevent cross-platform failures

Second Clone

- Now we will clone your new repo!
- First we have to make one

Creating a repo on GitHub

- Click the plus sign and then “New repository” on your profile:
 - Define name, description, public or private
 - Initialize with README (if you’re going to clone)
 - Please call it **sfdat22_work**
 - Nothing has happened to your local computer
 - This was done on GitHub, the website



Preview of what you're about to do

- Copy (“**clone**”) your new GitHub repo to your computer
- Make some file changes locally
- Save those changes locally (“**add**” and “**commit**” them)
- Update your GitHub repo with those changes (“**push**”)

Preview of what you're about to do

SUPER IMPORTANT:

- Make sure that you **LEAVE sfdat22** before cloning the new repo
- `cd ..`
- Never clone a git repo inside of another git repo!!!
 - Unless you wish to call upon the minions of the darkness.
In which case please contact your course advisor now

Making changes, checking your status

- Making changes:
 - Modify README.md in any text editor
 - Create a new file: `touch test.txt`
- Check your status:
 - `git status`
- File statuses (possibly color-coded):
 - Untracked (red)
 - Tracked and modified (red)
 - Staged for committing (green)
 - Committed
- Try this now! (take 1 minute)

```
[Sinans-MacBook-Pro:sfdat22 sinanozdemir$ touch test.txt
[Sinans-MacBook-Pro:sfdat22 sinanozdemir$ ls
README.md      notebooks      slides         test.txt
[Sinans-MacBook-Pro:sfdat22 sinanozdemir$ git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        test.txt

nothing added to commit but untracked files present (use "git add" to track)
Sinans-MacBook-Pro:sfdat22 sinanozdemir$
```

Note: Make sure you are in sfdat22_work and **NOT** sfdat22

Committing changes

- Stage changes for committing:
 - Add all “red” files: `git add .`
- Commit changes:
 - `git commit -m “message about commit”`
- Check your status again!
- Try this now! (take 3 minutes)

```
[Sinans-MacBook-Pro:sfdat22 sinanozdemir$ git add .  
[Sinans-MacBook-Pro:sfdat22 sinanozdemir$ git commit -m "Added test.txt"  
[master e7642ff] Added test.txt  
  1 file changed, 0 insertions(+), 0 deletions(-)  
  create mode 100644 test.txt  
[Sinans-MacBook-Pro:sfdat22 sinanozdemir$ git status  
On branch master  
Your branch is ahead of 'origin/master' by 3 commits.  
  (use "git push" to publish your local commits)  
nothing to commit, working directory clean  
Sinans-MacBook-Pro:sfdat22 sinanozdemir$
```

Pushing to GitHub

- Everything you've done to your cloned repo (so far) has been local
- You've been working in the “master” branch
- Push committed changes to GitHub:
 - Like syncing local file changes to Dropbox
 - `git push <remote> <branch>`
 - Often: `git push origin master`
- Refresh your GitHub repo to check!

```
Sinans-MacBook-Pro:sfdat22 sinanozdemir$ git push origin master
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 509 bytes | 0 bytes/s, done.
Total 5 (delta 2), reused 0 (delta 0)
To https://github.com/sinanuoazdemir/sfdat22.git
    d74267e..e7642ff  master -> master
Sinans-MacBook-Pro:sfdat22 sinanozdemir$
```

Quick recap of what you've done

- Created a repo on GitHub
- Cloned repo to your local computer (**git clone**)
 - Automatically sets up your “origin” remote
- Made two file changes
- Staged changes for committing (**git add**)
- Committed changes (**git commit**)
- Pushed changes to GitHub (**git push**)
- Inspected along the way (**git remote**, **git status**, **git log**)

Before you leave today

- Install [Git](#) on your machine
- Make a [Github](#) Profile on the web
- Create your own repo, call it “sfdat22” and clone it to your machine
- Clone the [class repo](#)

BREAK TIME

- Grab some coffee, you've earned it because you've learned it.

IV. Bonus Content

Git installation and setup

- Installation: tiny.cc/installgit
- Open Git Bash (Windows) or Terminal (Mac/Linux):
 - `git config --global user.name "YOUR FULL NAME"`
 - `git config --global user.email "YOUR EMAIL"`
- Use the same email address you used with your GitHub account
- Generate SSH keys (optional): tiny.cc/gitssh
 - More secure than HTTPS
 - Only necessary if HTTPS doesn't work for you

Checking your remotes

- A “remote alias” is a reference to a repo not on your local computer
 - Like a connection to your Dropbox account
- View remotes: `git remote -v`
- “origin” remote was set up by “git clone”
- Note: Remotes are repo-specific

Two ways to initialize Git

- Initialize on GitHub:
 - Create a repo on GitHub (with README)
 - Clone to your local machine
- Initialize locally:
 - Initialize Git in existing local directory: `git init`
 - Create a repo on GitHub (without README)
 - Add remote: `git remote add origin <URL>`

Deleting or moving a repo

- Deleting a GitHub repo:
 - Settings, then Delete
- Deleting a local repo:
 - Just delete the folder!
- Moving a local repo:
 - Just move the folder!

Excluding files from a repo

- Create a “.gitignore” file in your repo: **touch .gitignore**
- Specify exclusions, one per line:
 - Single files: pip-log.txt
 - All files with a matching extension: *.pyc
 - Directories: env/
- Templates: github.com/github/gitignore

Gists: lightweight repos

- You have access to Gist: gist.github.com
- Add one or more files
- Supports cloning, forking, commenting, committing
- Can be public or secret (not private)
- Useful for snippets, embedding, IPython nbviewer, etc.

Useful to learn next

- Working with branches
- Rolling back changes
- Resolving merge conflicts
- Fixing LF/CRLF issues