

A Guide to Git and Github Table of Content_version_2

```
GETTING STARTED
   Introduction
       Source Control Management
       Code Repositories
       What is Git?
      Local vs. Remote
      What is GitHub?
WORKING LOCALLY
   Creating Repositories
       git init
      The config File
      .gitignore
   Tracking Change
      Creating a commit
       git log
       Summary
   Branching
      What is Branching?
      Why Would I Want to Branch?
       Command Reference
       Command
       Purpose
REMOTE REPOSITORIES
   Connecting to GitHub Repo
   GitHub Authentications
   git push
   git pull
   git clone
   Summary
CONCLUSION
   Cheatsheets
```

GETTING STARTED

Introduction

Source Control Management

Source control Management is the process programmers follow to iterate through their projects or codebases.

Source control/Version control is the practice of tracking and managing changes to software code.

Code Repositories

repositories = directory and files

What is Git?

one of the most popular SCM tool dvcs

Local vs. Remote

local repository = located on your computer
remote repository = located on an other computer or server

What is GitHub?

is an ecosystem and user interface for software projects

WORKING LOCALLY

Creating Repositories

git init

git init # initializes a new reporitory by adding the .git directory Remember to create a README.md a LICENSE.md and a .gitignore directory for every git repo you're working with. Do not nest your repositories!

The config File

Location

/etc/gitconfig

.

.

Description

Contains settings that are system-wide and apply to all users and all if their repositories.

code sample

git config --system user.name 'bill'
saves user name to system wide
config file /etc/gitconfig

~/.gitconfig

- .
- .
- .

Configuration that is specific to your user account. This file overrides/etc/gitconfig git config --global user.name 'bill'
saves user name to user wide
config file ~/.gitconfig
.

REPO/.git/config

- •
- .
- •
- .

Configuration that is specific to a repository. this file overrides both of the

other files.

git config user.name "bill zachary"
saves user name current repo's
config file /.git/config

.

.gitignore

is a list of files and/or directories that you do not want included in your repository (private data such databases info...).

Remember to create a .gitignore file for every git repo you're working with

Tracking Change

Creating a commit

is a bundle of changes Any unstaged changes are not tracked by git!

git log

git log # history of commits in this repository commit hygiene is highly regarded is some teams so it's important to make sure each commit contains a logical grouping of changes (eg, a bug fix or a feature) with a clear commit message!

Summary

Command Description

git status

Run this command any time and often to check on the status of the files in the git repository.

git add

•

This command stages changed files, readying them to be wrapped into the next commit.

git commit

•

This command commits staged files, wrapping them into a commit. A historical record of commits is what we refer to as a codebase's version or commit history.

git log

View the repository's commit history.

Branching

What is Branching?

a branch is a copy of all the files in your codebase.

Why Would I Want to Branch?

to create a branch we fork it from another existing branch, likely main. If we decide the experiment is a sucess we can merge commits from the experimental branch into our main branch.

Command Reference

Command Purpose

git init

Creates a new reporsitory in the current directory. Generates a .git directory.

git config

Sets git configuration settings, such as author email and name.

git status

.

Shows the working directory as well as the staged changes. Run this command liberally to see which changes are ready for committing.

git add FILENAME

.

Stage file changes, which prepares them to be added to the repository.

git commit -m "message" Creates a commit from staged files.

Commits are the atomic unit in git that gets moved around

REMOTE REPOSITORIES

Connecting to GitHub Repo

```
    creating a remote repo on Github
    connecting my local repo to the Github remote repo:
    make sure that your local repo contains the README.md, LICENSE.md and .gitignore
    files and directory
    git remote add origin git@github.com:wguiraud/remote_repo_name.git
```

GitHub Authentications

```
TO DO LATER ON => https://launchschool.com/books/git/read/github#githubauth FOR NOW I STICK TO SSH.
```

git push

```
git push -u origin main # origin is an alias for the remote repo name
```

git pull

```
# checking remote repo for changes
git fetch

# checking what has changed
git diff main origin/main # main = local repository, origin/main = remote repo
=> @@ -0,0 +1 @@ # + represents added line from the repo specified second on
# the command line, while - represents deleted lines from the repo specified
# first.

# pulling from remote repo origin from the main branch
git pull --ff-only origin main # --ff = fast-forward merge to combine the changes
# (moves the history of the local branch forward to match the history fetched
# from the remote branch) -only = if git can't combine the changes with the ff
# merge it should abort the merge
```

git clone

```
# pulling down all the content of a remote repo
git clone <remote repo url> <local repo name>
```

Summary

Command

When to use

git init

Create a new local repository.

git remote add origin REMOTE_URL

Add an existing remote repo as a remote of existing local repo.

git clone

Pull down contents of existing remote repo into a new local repo, and add a remote ro the local repo pointing to remote repo.

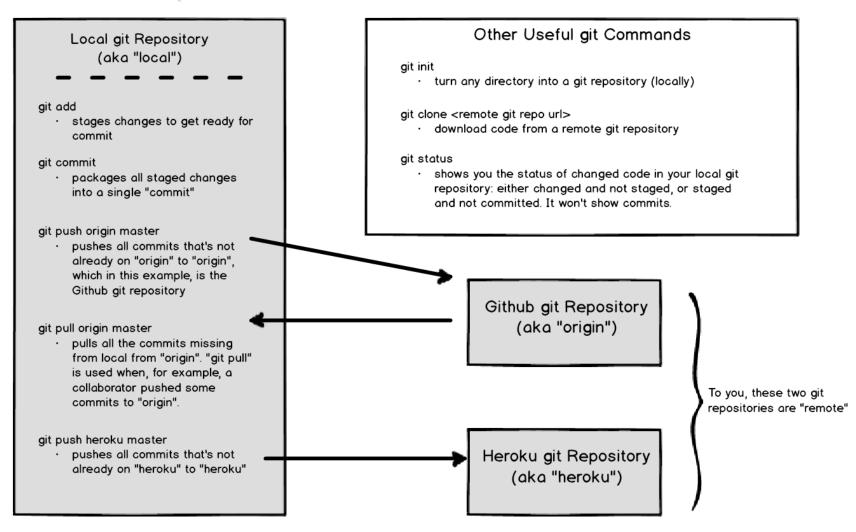
Let's rehash what just happened in this chapter:

- First, we created a repository on <u>github.com</u>, and chose not to initialize it with any files, leaving it a blank slate.
- Next, we made our local repo aware of our newly created remote repo with git remote
 add origin REMOTEURL.
- Then, we pushed some commits from our local repo to our remote GitHub repousing git push.
- We then made a modification on <u>github.com</u> directly, simulating a coworker pushing commits to our remote <u>github.com</u> repo.
- Finally, we pulled those commits down to our local repository using git pull, syncing up our local repo with our remote repo.
- We also learned what git clone is, and how you would use it to work on an existing git repository.

CONCLUSION

Cheatsheets







Create a git repository on your machine (aka, your "local")

you can turn any directory on your machine into a git repository by issuing "git init". Make sure not to nest git repositories. You are allowed to create other folders within a git repository, just not another ait repository.

git clone <remote git url>

· you can download an existing project with "git clone", and this will also be a git project.

This directory is now your project directory and also the local git repository for your project. You can create new files and folders in this git repository, but don't create another git repository under it.

You only need to do this one time to set up this git repository.

2. Stage Changes

git add <path/to/file>

now, from your git repository (also your project directory), you can "stage changes" to ready them for commit. These are changes you make to any file or folder within the git repository, including, modifying a file contents, adding new files/ folders, deleting files/folders.

There are some shortcuts you can use, such as "git add ." or "git add -u", but in the beginning, practice by "git add" every

You can use another helpful git command, git status, to see all modified but unstaged files, and also all staged files. You should get familiar with "git status" and use it extensively.

3. Package Staged Changes Into A Commit

git commit -m "a commit message"

this will package all the "staged" files from step #2 into a commit, along with a message.

In git, commits are what you push and pull around to/from other repositories. Commits will not show up with "git status". To see a list of commits in the repository, you can use git log.

Push Commits To Remote Repository

git push origin master

this will push all commits that's not yet on "origin" to "origin".

"origin" in the above code example is the alias to a remote git repository, usually an associated repository hosted on

Where did "orgin" come from? You have to first tell your local git repository that from now on, we're going to alias "origin" to the url of a remote git repository. So in order to push/pull to/from "origin", you had to first have issued git remote add origin <url of remote git repository>