GitHub Intro

Goals

- Compare and contrast Git and GitHub
- List use cases for GitHub
- Get started with GitHub!

What is GitHub?

GitHub is a web-based Git repository hosting service.

Simply put, it is a tool that enables collaboration by hosting shared Git repositories that teams of developers can all contribute to.

While GitHub uses Git, the functionality it provides is **VERY** different from Git so make sure you understand that Git and GitHub are not the same thing.

In short, Git is a Version Control System. GitHub is an online platform for hosting and sharing code, text files and even more complex file formats.

Why use GitHub?

It provides a great way for you to store your code in a remote location

It's a fantastic way to collaborate with other developers both privately and publicly

Many large open source projects are hosted on GitHub, which makes it very easy to examine the code both on GitHub and locally

In the next couple of chapters, we will learn how to move code from our local repository to a remote repository on GitHub using the ***push*** command

Who uses GitHub?

Here are some projects you may have heard of that are hosted there:

- React - https://github.com/facebook/react <https://github.com/facebook/react>
- Node.js - https://github.com/nodejs/node <https://github.com/nodejs/node>
- Angular - https://github.com/angular/angular <https://github.com/angular/angular>
- Ruby on Rails - https://github.com/rails/rails <https://github.com/rails/rails>
- Twitter Bootstrap - https://github.com/twbs/bootstrap <https://github.com/twbs/bootstrap>
- jQuery - https://github.com/jquery/jquery <https://github.com/jquery/jquery>
- Homebrew - https://github.com/Homebrew/brew <https://github.com/Homebrew/brew>

Getting started with GitHub

If you don't have an account with GitHub yet, head to https://github.com/ <https://github.com/> and create an account

Be sure to use whatever email address is in your ***.gitconfig*** for your email address when you sign up with GitHub.

If you'd rather sign up with a different email address, change your ***.gitconfig*** accordingly.

You'll run into some minor annoyances if there's a mismatch between the email address in your GitHub profile and the email address in your ***.gitconfig***.

Creating Remote Repositories on GitHub

- Navigate to https://www.github.com/new <https://www.github.com/new>
- After naming and creating a new repository click ***create repository***
- Follow the second block of instructions for pushing an existing repository from the command line

What are we doing here?

- Adding a ***remote***
- Calling it "origin"

What's a remote?

- It's a nickname for a URL where your repository lives!
- Instead of typing/remembering the entire URL, we give it a nickname
- By default this nickname is ***'origin'***
- ***git remote add NAME_OF_REMOTE URL_FOR_REPOSITORY***
  - ***NAME_OF_REMOTE***: origin
  - ***URL_FOR_REPOSITORY***: https://github.com/YOUR_USERNAME/YOUR_REPO.git <https://github.com/YOUR_USERNAME/YOUR_REPO.git>

git remote add

This command tells our local repository about a remote repository located somewhere.

The location of our remote repository is ***https://github.com/YOUR_USERNAME/YOUR_REPO.git***.

git push origin

Now if we want to send our code to GitHub we could type in

git push https://github.com/YOUR_USERNAME/YOUR_REPO.git master

but typing that whole URL is quite a pain.

Instead, we give the URL a nickname (also called an alias) - “origin”

- Once the alias is set up we can simply type
- ***git push origin master***
- This will send the code from our ***master*** branch to this remote repository.
- To see where your remotes are pointing to locally, type ***git remote -v***.
- This will display both the alias and the remote url

If you need to remove a remote you can use ***git remote rm NAME_OF_REMOTE***

Pushing Your Code

The command we use is ***git push NAME_OF_REMOTE NAME_OF_BRANCH***

Setting the upstream

```
git push -u origin master
```

Now we can send our code from a local repository to our remote repository (which we aliased to ***origin*** in the previous command).

The ***-u*** flag allows us in the future to only have to type ***git push*** instead of ***git push origin master***.

Pushing code up to GitHub

Now when you type in ***git push***, you will be prompted to enter your username and password for GitHub. While that is fine once or twice, it becomes quite a nuisance if you are pushing or pulling (retrieving code) frequently.

It would be nice if we could establish some trust between our computer and GitHub so that when we run ***git push*** or ***git pull***, GitHub does not need to authenticate us. To do that we are going to create an SSH key.

Practice

- Generate some code files on your machine and set up a local repository using ***git init .***
- Create a remote repository on github to push your code to
- Use ***‘git remote add’*** to point the local repo to the remote repo and set up an alias (origin)
- Add and commit your code to generate a master branch
- Push your code to the remote repo using ***‘git push origin master’***
- Go back to github and refresh the page, you should be able to see your code
- Change some code and add and commit a second time
- This time set the upstream branch

- You should now be able to push your code with ***'git push'*** in the future

More Practice

- Create another remote repo on github
- Point the existing local repo to the new remote repo and set up an alias
  - We have shown how to view, add and remove the remote
  - Research ***git remote set-url*** if you are curious how update the remote with one line of code
- Push your code to the second remote repo
- Examine both the repos on Github that you have pushed code to
  - Notice the new repo has all of the commit history
  - The commit history belongs to the local git repo
  - The remote github repo just provides a interface for people to share code