GITHUB BASICS

APRIL 2022

YONATAN BENEZRA

In this lesson, we will discuss the following topics:

- git/hub

- Pre-Install
- Installing Git
- Creating a Local Repository
- Github
- Cloning
- Cheat Sheet

What is Git? It is a version-control.

It is a way to save our lines of code to one place on our local computer.
Github can be used to save your code to the web as well.

There is a lot more to Github, which you can read about here.

In this lesson, we will:

- Install Git and create our first local repository.
- Create a repo on Github.
- Upload our local repo files to Github.
- Make changes.

Installing Git:

Download it from here and install it using the default settings.

Important installation instructions:

Start the installation and wait until you get to the selecting page.

Setup First - Let's open the cmd.

This is called a Terminal.
After opening the editor, let's start using it! Note - every line automatically starts with a dollar sign, which is necessary, but is automatic in this editor. To make sure Git is installed correctly, type:

```
git --version
```

If you see a Git version, that means it is installed correctly. To set up Git, we need to configure our username and email. You can do so by running the following commands:

```
git config --global user.name "Your username"
git config --global user.email "your_email@gmaill.com"
```

Congratulations! Our next step is to create a local repository.

Creating a Local Repository:

A repository, or Repo, is where our code lives, both on our computer and on a host computer (such as Github, which we will use later on).
We will start by creating the local repository, which is located on our computer.

Let's enter our CSS-HTML project directory. Type the following in your editor:

```
ls
```

See which folders you have in the current directory and enter the folder which contains your project.

For example, in order to reach my project folder, I will type the following:

```
ls
cd desktop
cd code
cd week1
cd CSS-HTML project
```

Now we want Git to keep track of all the files inside our directory.

***Note:**
A repository is just a storage space where Git can keep track of all files inside it.

To make this happen, we need to write the following code. Make sure you are in the directory which contains your project folders.

```
git init
```

You should see a message that says something like "Initialized empty Git repository in..."

This means Git is now keeping track of the folders. This step must be completed only once for each directory.

We can now add files, update them, and update the remote repository.

Github:

I think we are ready to move on to our Git project which is located locally on github.com.
This way other developers can see it, contribute to it, fork it, clone it, and most importantly, it will serve as a backup if needed.

Creating a repository on GitHub

- Go to github.com and log in to your account.
- Go to your profile and click on the "Repositories" tab.
- Click on the big green button that says "New".
- Give your project a name.
- If you cannot think of a name, name the project: "CSS-Html"
- Lastly, click "Create repository".

The instructions on the next screen can be a bit confusing so stick to these instructions and I promise it will work out. In your editor, the note will make sure it is in the Projects directory. Enter the following command:

```
git status
```

This will show you the status of all changed files in this directory.
There are several additional steps that must be completed:

- Add our changed files to our Git.
- Save these changes (commit) with an explanation.
- Add a remote origin to Github.
- Push our local Git changes to our GitHub.

Step 1:

When you typed the command "git status", you probably saw some red lines. Each of these lines represents a changed file. Adding those files to our Git will turn the lines green! Let's do this by entering the following command in the editor:

```
git add .
```

We are now adding to Git everything that has changed. Alternatively, you can add a certain individual file by swapping the "." for a file name, as follows:

```
git add index.html
```

Step 2:

After adding the changed files in step 1, we would now like to commit to the changes, along with an explanation of what we are committing to. We do so by using the commit command, as follows:

```
git commit -m "Initial commit"
```

This commits to all the changes we added in step 1.

Step 3:

To connect between our local Git and our GitHub repo, we must return to our repo in GitHub. Copy the URL at the top, head back to your command line, and make sure you are in your project folder.

Run the below command, only swap "YOUR_URL" for the URL you just copied.

```
git remote add origin YOUR_URL
```

This command just added a reference to a "remote repository" to our local repository.

Step 4:

In our final step, after connecting between our local repo and our GitHub repo, we now need to push all the changes we committed with the following line of code:

```
git push origin master
```

The above command is telling Git to "push our project to the origin repo's master branch". We'll talk more about branches later on. Now, if you go back to Github and refresh the repos page, you will see your files!

Congratulations!!

Now let's change something in our code and see how we update it in our GitHub. First things first - go to your index.html and change one word in it. Afterward, enter the following command:

```
git status
```

Once again, you should see a red line that represents the file changes (index.html). The following steps are very important.

Enter the following three commands that we have learned:

```
git add .
git commit -m "YOUR_EXPLANATION"
git push origin master
```

Check your Github... Amazing!

Cloning

Click this link to navigate to the repo.

To fork this repo, you must:

- Click on the button in the top right that says "fork"...
- Then, select your profile.
- GitHub has now made a copy of our repository on your profile.
- Now, click "Clone or download", and copy the URL that is displayed.
- Next, open your terminal and navigate to wherever you're storing your code - for example code/week1/ and type:

```
git clone <PasteURLHere>
```

(Of course, you need to replace "PasteURLHere" with the URL you just copied):

Run the command (press enter). This will create a new folder on your computer, called Fork.
cd (enter) into that folder as usual and open it in vs code to start working with it!

When you clone a GitHub repo, it will also set up a remote called "origin". Therefore, there is no need to do the whole Git remote add origin ... thing.

By the way, you do not always have to fork repositories.
If you're working on an existing repository and contributing to it, it's enough to just clone it so you have local access to it.

Next, you can make your changes and push them back to the original repo.
Generally, we will only fork repos if we want to create something entirely new based on something existing.