Week 1

Git and Github

What is Git?

Git is a version control system which lets you track changes you make to your files over time. With Git, you can revert to various states of your files (like a time traveling machine). You can also make a copy of your file, make changes to that copy, and then merge these changes to the original copy.

How to configure Git

After Installing git we run this command to verify this: `git --version`. This shows you the current version installed on PC.

The next thing you'll need to do is to set your username and email address. Git will use this information to identify who made specific changes to files.
To set your username, type and execute these commands: `git config --global user.name "YOUR_USERNAME"` and `git config --global user.email "YOUR_EMAIL"`. Just make sure to replace `"YOUR_USERNAME"` and `"YOUR_EMAIL"` with the values you choose.

How to Create and Initialize a Project in Git

to initialize your project, simply run `git init`. This will tell Git to get ready to start watching your files for every change that occurs. It looks like this:

git init

The first line has information about my PC and the path to where the folder exists.
The second line is the command `git init`, and
the third line is the response sent back telling me that my repository (repo) has been initialized. It is considered empty because we have not told Git what files to track.
A repository is just another way to define a project being watched/tracked by Git.

What is GitHub?

GitHub is an online hosting service for Git repositories. Imagine working on a project at home and while you are away, maybe at a friend's place, you suddenly remember the solution to a code error that has kept you restless for days.
You cannot make these changes because your PC is not with you. But if you have your project hosted on GitHub, you can access and download that project with a command on whatever computer you have access to. Then you can make your changes and push the latest version back to GitHub.
In summary, GitHub lets you store your repo on their platform. Another awesome feature that comes with GitHub is the ability to collaborate with other developers from any location.

Now that we have created and initialized our project locally, let's push it to GitHub.

How to push a repository to GitHub

Step 1 – Create a GitHub account

Step 2 – Create a repository

Step 3 – Add and commit file(s)

Committed state

A file is in the **committed** state when all the changes made to the file have been saved in the local repo. Files in the committed stage are files ready to be pushed to the remote repo (on GitHub).

Modified state

A file in the **modified** state has some changes made to it but it's not yet saved. This means that the state of the file has been altered from its previous state in the committed state.

Staged state

A file in the **staged** state means it is ready to be committed. In this state, all necessary changes have been made so the next step is to move the file to the commit state.

How to add files in Git

When we first initialized our project, the file was not being tracked by Git. To do that, we use this command `git add .` The period or dot that comes after `add` means all the files that exist in the repository. If you want to add a specific file, maybe one named `about.txt`, you use `git add about.txt`.

Now our file is in the staged state. You will not get a response after this command, but to know what state your file is in, you can run the `git status` command.

How to commit files in Git

The next state for a file after the staged state is the committed state. To commit our file, we use the `git commit -m "first commit"` command.

The first part of the command `git commit` tells Git that all the files staged are ready to be committed so it is time to take a snapshot. The second part `-m "first commit"` is the commit message. `-m` is shorthand for message while the text inside the parenthesis is the commit message.

After executing this command, you should get a response similar to this:

git commit

Now our file is in the committed state.

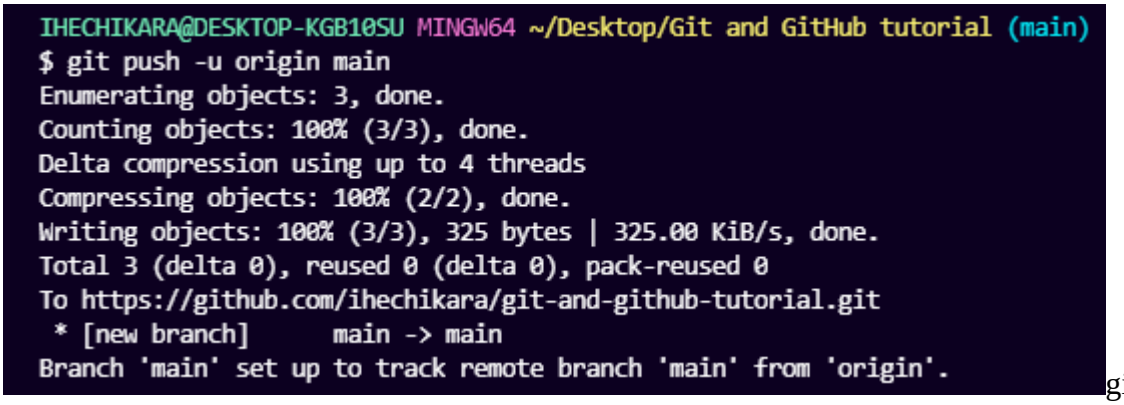
Step 4 – Push the repository to GitHub

After you create the repo, you should be redirected to a page that tells you how to create a repo locally or push an existing one.

In our case, the project already exists locally so we will use commands in the "…or push an existing repository from the command line" section. These are the commands:

```
git remote add origin https://github.com/ihechikara/git-and-github-tutorial.git
git branch -M main
git push -u origin main
```

The first command `git remote add origin` https://github.com/ihechikara/git-and-github-tutorial.git creates a connection between your local repo and the remote repo on Github. The second command `git branch -M main` changes your main branch's name to "main". The default branch might be created as "master", but "main" is the standard name for this repo now. There is usually no response here.
The last command `git push -u origin main` pushes your repo from your local device to GitHub. You should get a response similar to this:



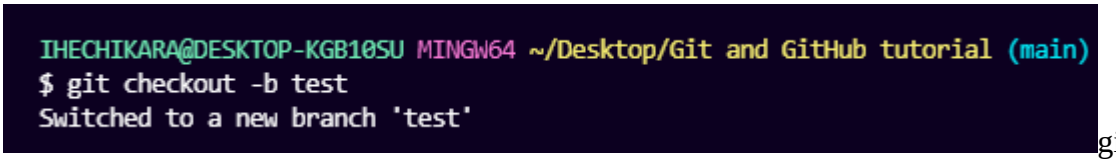
git push

How to Use Branches in Git

With branches, you can create a copy of a file you would like to work on without messing up the original copy. You can either merge these changes to the original copy or just let the branch remain independent.

To create a new branch, run this command: `git checkout -b test`.

`checkout` tells Git it is supposed to switch to a new branch. `-b` tells Git to create a new branch. `test` is the name of the branch to be created and switched to. Here is the response you should get:

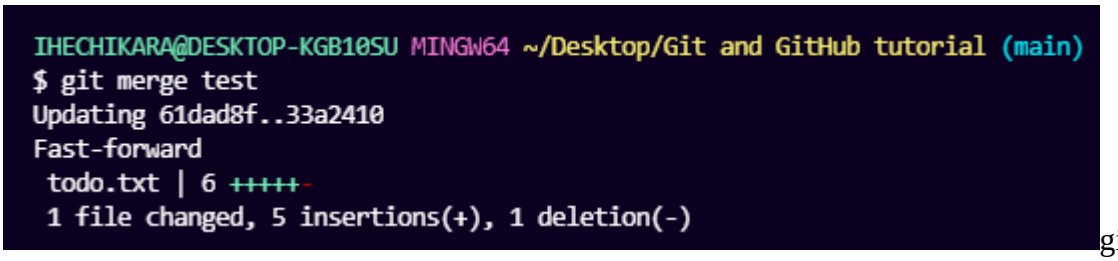


git checkout -b

After committing your test branch, switch back to the main branch by running this command: `git checkout main`.

You can check all the branches that exist in your repo by running the `git branch` command.

Now we can merge the changes we made in the test branch into the main branch by running `git merge test`. At this point, you will see all the changes made in the test branch reflected on the main branch. You should also receive a response similar to this:



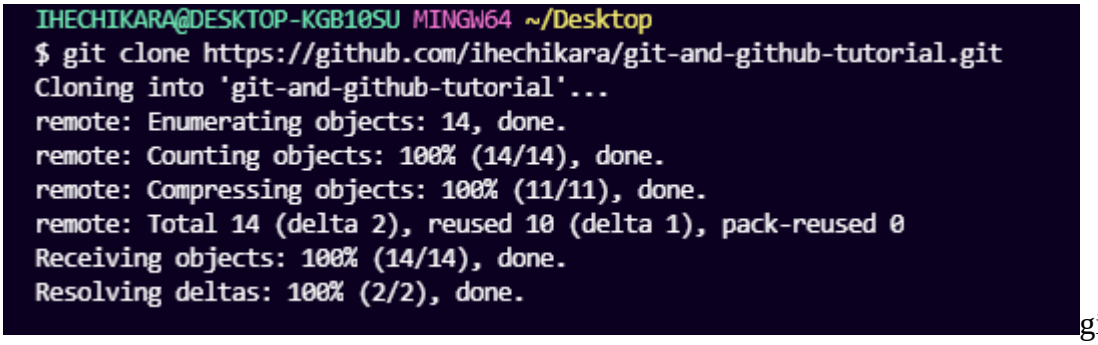
git merge

If you go on to push your repo to GitHub, you'll see that the test branch will not be pushed. It will only remain in your local repo. If you would like to push your test branch, switch to the branch using `git checkout test` and then run `git push -u origin test`.

How to Pull a Repository in Git

To pull in Git means to clone a remote repository's current state into your computer/repository. This comes in handy when you want to work on your repo from a different computer or when you are contributing to an open source project online.

run `git clone YOUR_HTTPS_URL`. This command pulls the remote repository into your local computer in a folder



git clone

freecodecamp tutroial as a reference