Data.Trek 2020: Git Tutorial

# Pre-requirements

* On Windows :
  + Git bash (more user-friendly than your windows terminal)
  + Git
* On Linux:
  + Git

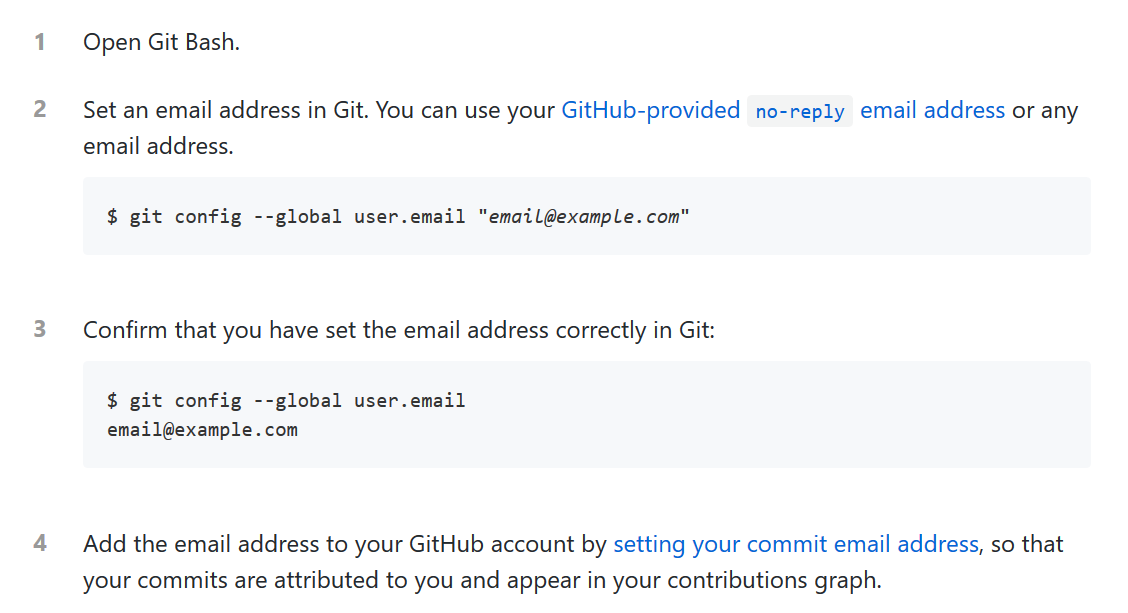
## Create a GitHub account

* Sign up on <https://github.com/>

## Configure your username and email on Git

* Note: *--global* will set this username and email for all your Git repository

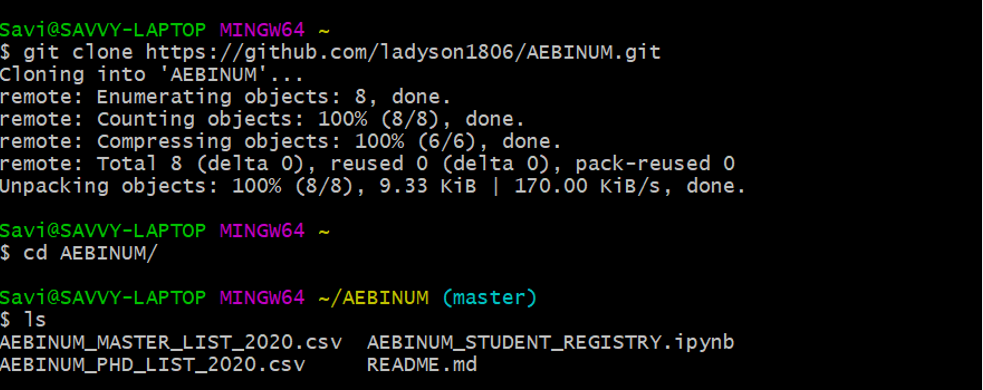




## Create your Git repo and clone it into your local computer

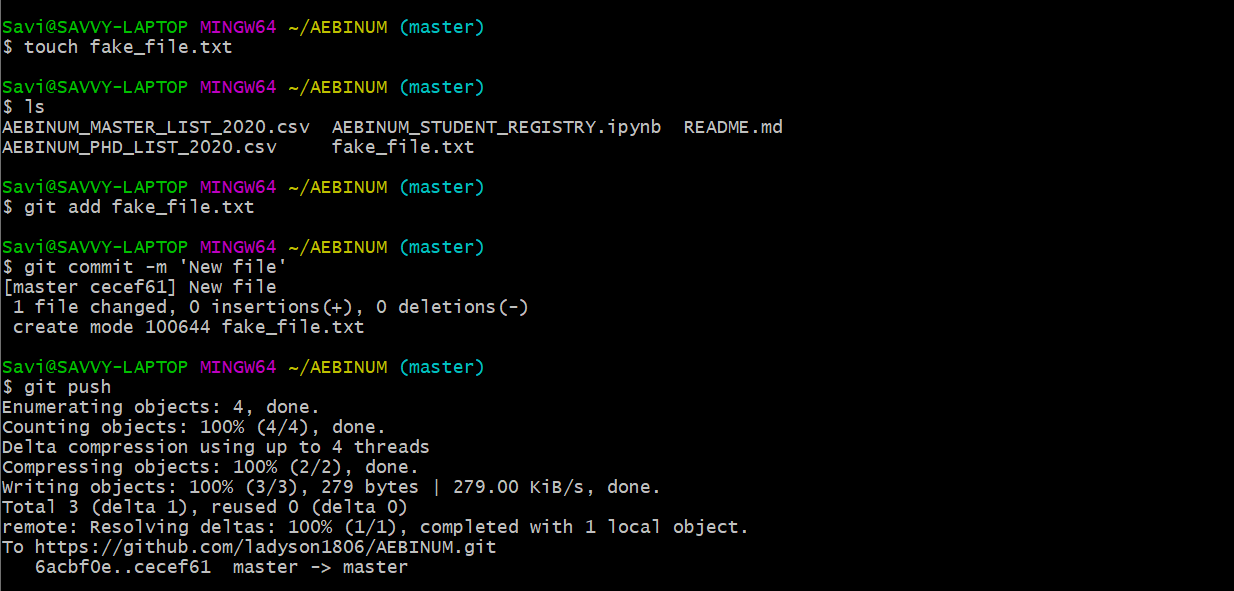
* Follow the guidelines
  + Here: <https://help.github.com/en/github/getting-started-with-github/create-a-repo>
* Stop the reading at this line: “Congratulations! You've successfully created your first repository and initialized it with a *README* file.”
  + The next part is about making a commit from the web interface
* **Clone** your repository on your computer

1. Open Git Bash
2. Write `**cd TO/YOUR/PATH/FOLDER**`
3. Write `**git clone** [**http://www.github.com/your\_repository\_url`**](http://www.github.com/your_repository_url%60)
   * + Your will collect this link on the main page of your repository by clicking on the “Clone/Download button”
4. Enter in the folder using `**cd**` and check if the content of your report was created with `**ls**`
   * + You should have only an empty README.md file



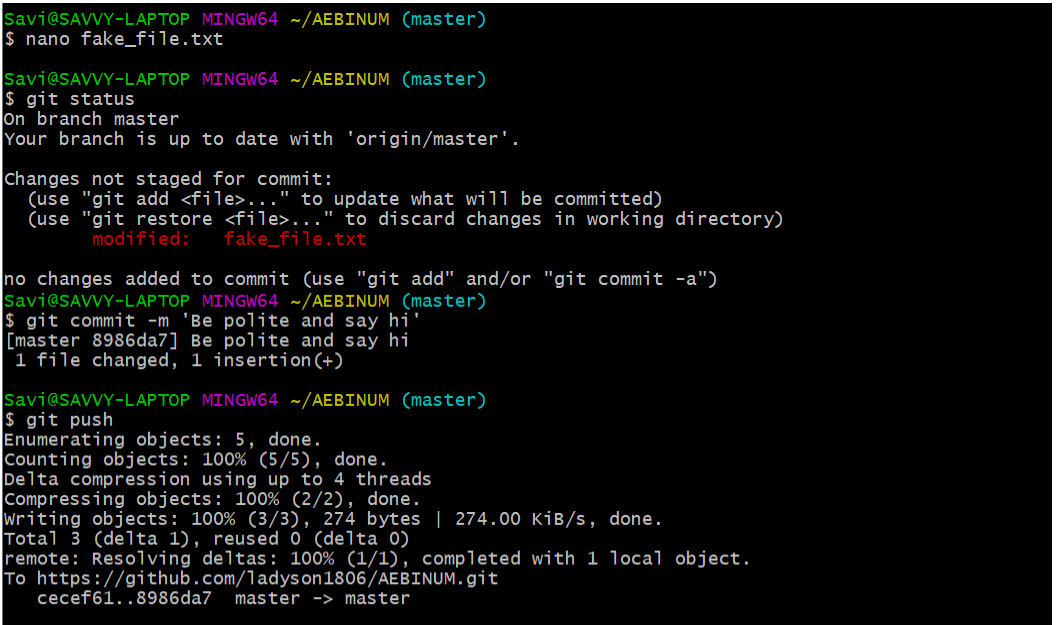
## Add a new file into your repository and update your Git repo

1. Create a file in your folder
   * In my example, it will be an empty txt file : fake\_file.txt
2. Write **`git add fake\_file.txt**` will “warn” your git that you want to commit a change
3. Write `**git commit -m “New file”**` will add a short description related to your commit
4. Write `**git push**` to update your git repo



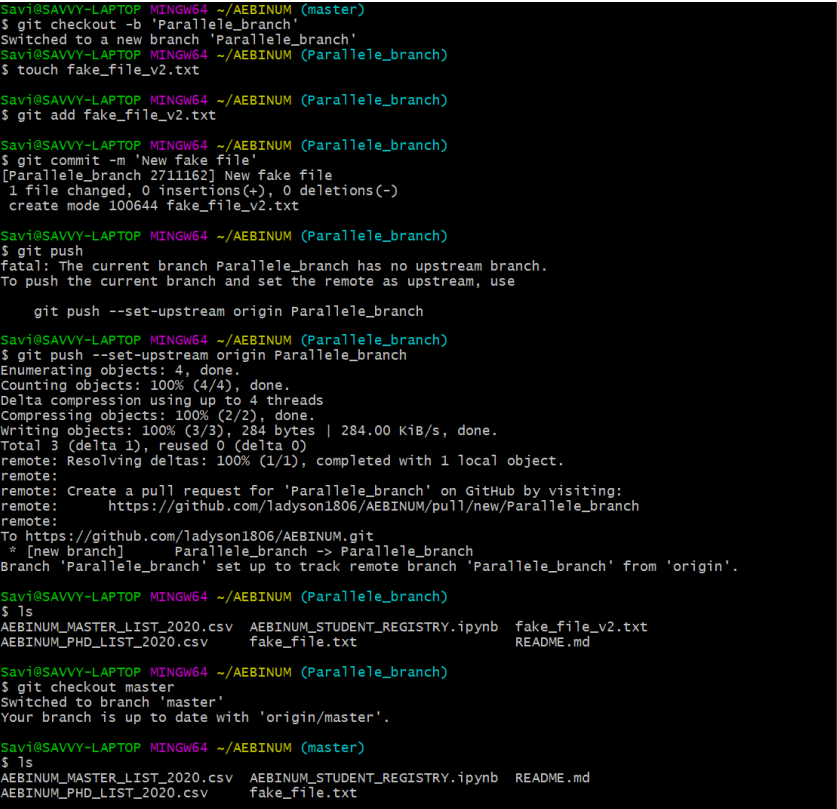
Notes:

* Always think to commit if you are modifying a file! (That’s make a historic of your changes aka. versioning)
  + Never forget this order: **add / commit / push**
* `**git status**` will give you an overview of what you need to commit
  + If you are working with someone, it is important to always have the last recent version of your Git repo
* ‘**git pull**’ will help you to have the last version of the Git repo from Github (not show in my example)
  + If you have something new in your Git repo (either because you add something through the web interface or someone else add something in the Git repo)
  + If nothing changed, it will tell you that your repo is up to date



## Create and work in a branch

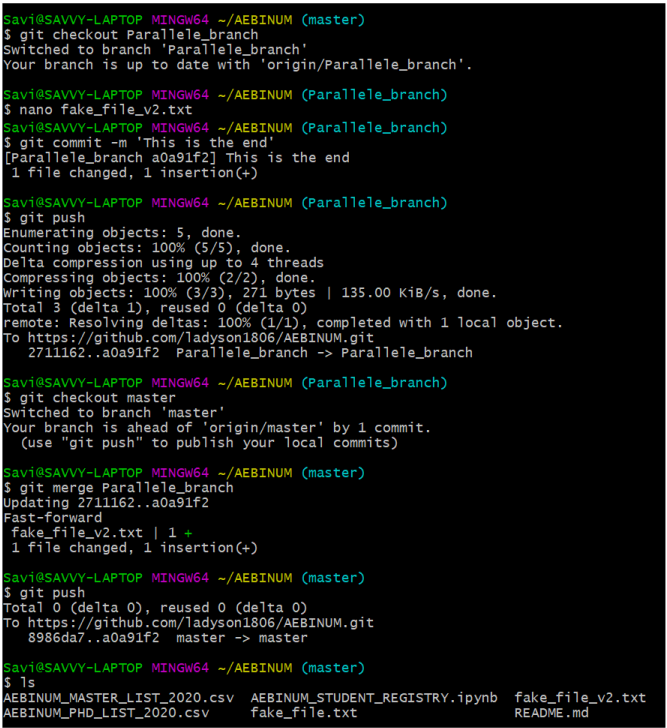
* Creating a branch is useful when you want to make some tests without modifying your master branch (the main one)
* In this example, I will create a branch called Parallele\_branch and I will add a new file in this branch
  + You can directly create a branch and be placed in it by writing **`git checkout -b Parallele\_branch`**
  + To switch between your different branches, you will the write the **command `git checkout BRANCHNAME (here master or Parallele\_branch)`**
* If you are working in team, be sure to create your own branch to not overwrite the code of your team mate!



Have you seen the differences between branches??

## Merge the content of a branch

* If the file you add in your second branch will be need in the final version of your code, you need to merge to the master branch



Notes:

* To delete a branch, write **`git checkout -d BRANCHNAME’**
  + Avoid deleting your master branch, tho’

**THE END!**

# References

* Github documentation
* <https://help.github.com/en/github/using-git/setting-your-username-in-git>
* <https://help.github.com/en/github/setting-up-and-managing-your-github-user-account/setting-your-commit-email-address>

# Acknowledgments

* Thanks to Francis, Gracielle and Gabriel for the Git Demonstration on Zoom! It helped me a lot to write this summary tutorial 😊