**gitGit Instructions**

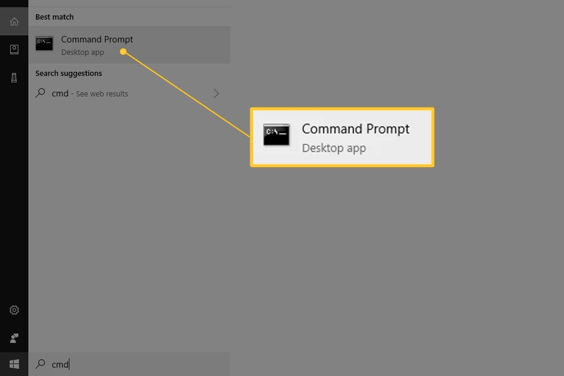
**WHAT IS GIT?**

*Git* is a version control system used for coordinating work among programmers. **Git allows** a team of people to work on a task together, tracks the changes each member makes to files and creates a record of what has been done, which later allows reverting to previous versions when needed.

**COMMAND LINE INTERPRETER**

**Windows:**

* *Type "command prompt” or ”cmd" into the Start menu.*

**

**Mac OS:**

* *Click on the Spotlight button in the menu bar (looks like a magnifying glass)*
* *Type “terminal”*

**

**Command Line Interpreter:** Basic Commands

|  |  |  |
| --- | --- | --- |
| **Command** | **Description** | **Sample Use** |
| **cd** | ”Change Directory" or, in simpler terms, change which folder we are in | • **cd desktop**  • **cd downloads** |
| **ls** | It shows everything in a folder. | • **ls DIR /W** |
| **clear** | It "clears" the screen | • **clear CLS** |
| **mkdir** | "Make Directory" or simply, make a new folder. | • **mkdir newFolder**  • **mkdir “X”** |
| **cd ..** | Will move the user up one directory. So, if you are in the .../myPictures/RoadTrip directory, it will take you to .../myPictures directory. | • **cd ..** |

**Breakout Room Practice Instructions**

1. Go to you command prompt (or terminal)
2. Go to your desktop
3. Create a folder named “My Project Folder”
4. Put/Copy some files in the My Project Folder (manually)
5. Print/Show the files in the “My Project Folder”
6. Clear the screen
7. Use the cd.. command

**SETTING UP GIT AND GITHUB**

**Git Installation:**

To be able to use Git, you will first need to install it on your computer:

* Windows:<https://git-scm.com/download/win>
* Mac:<https://git-scm.com/download/mac> Or <https://sourceforge.net/projects/git-osx-installer/>

After installation, type the following command on your command window (for windows) or terminal (for Mac).

* git --version

The output will be something like this:

* git version 2.28.0

If Git is not installed, you will get the following error:

* -bash: git: command not found

**GitHub Account Creation:**

Go to: <https://github.com/>

**Connecting your local computer with GitHub**

* <https://syntaxbytetutorials.com/add-a-github-ssh-key-on-windows/>
* COPY-PASTE the following lines one by one on to the command prompt
  + ssh-keygen -t rsa -b 4096 -C "[XYZ@example.com](mailto:example@example.com)"
  + eval $(ssh-agent -s)
  + ssh-add ~/.ssh/id\_rsa
  + cat < ~/.ssh/id\_rsa.pub

**Let the local repository know where to connect**

* First Create Personal Access Token on GitHub ([**Link**](https://stackoverflow.com/questions/68775869/support-for-password-authentication-was-removed-please-use-a-personal-access-to)).
  + From your GitHub account, go to
  + Settings => Developer Settings => Personal Access Token => Generate New Token (Give your password) => Fillup the form => click Generate token => Copy the generated Token, it will be something like ghp\_sFhFsSHhTzMDreGRLjmks4Tzuzgthdvfsrta
* sudo git remote add origin **https://github.com/username/directory-name.git**
* sudo git remote set-url origin https://**<token>**@github.com/username/directory-name.git
* ~~sudo git branch -M main~~
* ~~sudo git push -u origin main~~

**SOME GIT COMMANDS**:

There are numerous Git commands and this sheet presents a set of the most important commands that you will need in your job. To see all the available commands, you can always type:

**git help --all command**

Before you work on a project/task, you need to create an empty Git repo in the specified directory. In your terminal or command window, go to the folder that you want to create the git repository and type:

**git init**

This will initialize the existing directory as a Git repository and create a .git subdirectory. Now, it is time to add files to this repository. Let’s say we are working on a python project and we want to add the index.py file to the repository: You first step would be to “stage” the file on git, which can be done with the following command:  
  
 **git add index.py**

Note that we haven’t put anything on the Git repository yet. We just staged them on Git, which means that they are now the candidates to be pushed to Git. There is one more step before we push our changes: Adding a note for the changes we have made.

gittest

**git commit -m "added index.py file for managing the X files."**

Adding detailed and descriptive messages for git is very important given that many people work on the same project and we want everybody to have a clear understanding of the changes we are committing. Now we are ready to

**GIT/GITHUB IMPLEMENTATION STEPS**

1. **Create** a project on your GitHub and make sure you have a README file.
2. **Clone** your project from your Github to your computer. To do that, go to the git bash (or terminal on mac) on your computer and type the following command:

git clone [https://github.com/**username**/**directory-name**.git](https://github.com/username/directory-name.git)

git clone https://github.com/mehmet-renastech/Oct16-Project.git

1. **Connect your computer to your remote GitHub repository**:

git remote add origin [https://github.com/**username**/**directory-name**.git](https://github.com/username/directory-name.git)

git remote set-url origin https://**<token>**@github.com/**username**/**directory-name**.git

1. **Check the status of your repository.** This command is your best friend. It displays the state of the working directory and helps you view all the files which are going to be staged to the first commit. Use it very often (almost after every time you write a command). Make sure you read the output carefully.

git status

1. **Modify and add files to the new local repository.** This command “stages” your files on git, which means that they are now the candidates to be pushed to GitHub (or your remote repository)

git add **<fileName>**   *(this command will stage only one file)*

git add .      *(this command will stage all the new or modified files)*

1. **Commit your changes with a message:** Note that we haven’t put anything on the Git repository yet. We just staged them on Git, which means that they are now the candidates to be pushed to Git. Now, we will write  a detailed and descriptive message for the changes we made. Now we are ready to push our changes.

git commit -m "**<Some message>**"

1. **Push your changes to GitHub:** This is the final step: This will push your local content to GitHub.

git push -u origin **<branch\_name>**

git push -u origin main

**FORKING AN EXISTING PROJECT**

If you want to use an existing GitHub project **Fork** a project. Never copy someone else’s work without giving proper credits (otherwise, it is simply stealing) “Fork” the project, which will create a copy on your GitHub while still giving the credits to the original owner You can make any changes after your “fork” a project.

Graphical user interface, text, application, email, website

Description automatically generated

<https://github.com/arjunmann73/Data-Analytics-Projects>

After you fork the project, you can follow the steps above (except step 1). Also, you don’t need to use the following command because you already have a connection to the GitHub repository:

DON’T USE:  git remote add origin <https://github.com/username/directory-name.git>

However, you still need to use the following command to set the origin with your token:

git remote set-url origin https://**<token>**@github.com/username/directory-name.git

https://github.com/Kishp92/Bootcamp.Info.git  
  
  
ghp\_WWGqVta88hrF9yCBsNziJ2RKXvIMJE0wnXmf  
  
  
https://ghp\_WWGqVta88hrF9yCBsNziJ2RKXvIMJE0wnXmf@github.com/Kishp92/Bootcamp.Info.git

**GIT BRANCHING STEPS**

1. **Check the existing branch(es):**

git branch

1. **Create a new branch**

git branch myTestBranch

1. **Switch to a specific branch**

git checkout myTestBranch

1. **Add/Commit as usual**
2. **Push:** Note that the branch name is not main anymore!

git push -u origin <branch\_name>

git push -u origin myTestBranch

**GIT MERGING STEPS**

Suppose you’ve decided that your changes on the myTestBranch branch are complete and ready to be merged into your main branch:

On GitHub

**ADD COLLABORATORS**

On GitHub

If you want to work on the same branch [Link](https://stackoverflow.com/questions/1783405/how-do-i-check-out-a-remote-git-branch)

git fetch origin

git branch -v -a

git checkout -b <branch\_name> origin/<branch\_name>

**SHOW COMMIT HISTORY on GITHUB**

**How do I revert a Git repository to a previous commit?** [**LINK**](https://stackoverflow.com/questions/4114095/how-do-i-revert-a-git-repository-to-a-previous-commit)

Also git log

git reset --hard <commidId> && git push --force

This will reset the branch to the specific commit and then will upload the remote server with the same commits as you have in local.

Be careful with the --force flag as it removes all the subsequent commits after the selected commit without the option to recover them.

**ADVANCED TOPICS**

If you want to work on the same branch [Link](https://stackoverflow.com/questions/1783405/how-do-i-check-out-a-remote-git-branch)

git fetch origin

git branch -v -a

git checkout -b <branch\_name> origin/<branch\_name>

**Stashing**

Stash your changes

git stash

Stash List

git stash list

Revert a stash

Git stash apply 0

**Great Resource:**

<https://www.w3schools.com/git/default.asp?remote=github>

**Exercise:**

<https://www.w3schools.com/git/exercise.asp?filename=exercise_getstarted1>

**Quiz:**

<https://www.w3schools.com/quiztest/quiztest.asp?qtest=GIT>

TOKEN:

ghp\_WWGqVta88hrF9yCBsNziJ2RKXvIMJE0wnXmf

CLONE:

<https://github.com/Kishp92/Oct16_Project.git>

git remote set-url origin:

gi

Project 2

[https://**<token>**@github.com/**username**/**directory-name**.git](https://%3ctoken%3e@github.com/username/directory-name.git)