# Git Commands Manual

This manual provides a simple and clear explanation of essential Git commands in sequential order. It's designed to help beginners quickly understand and use Git effectively.

## 1. git --version

Checks the current version of Git installed on your system.

## 2. mkdir directoryName

Creates a local directory on your PC.

## 3. cd directoryName

Moves you into a directory.

## 4. ls

Lists all folders/files in the current directory.

## 5. touch fileName.extensionName

Creates a file in the current directory with the specified extension, e.g., touch demo.txt.

## 6. git init

Initializes a new Git repository in the current directory. It sets up a .git folder, which contains all the metadata and configurations required for Git to track changes in your files.

## 7. git status

Shows the current status of your repository, displaying changes in your files and helping you see what needs to be added or committed.

## 8. git add fileName.extensionName OR . (for all files in directory)

Adds the specified file to the staging area, preparing it to be committed.

## 9. git commit -m "message"

Saves your staged changes into the repository. The -m option allows you to include a message summarizing the changes.

## 10. git config --global user.username yourUsername

Sets your username for Git globally (applies to all projects). This information identifies who made the commits.

## 11. git config --global user.email yourEmail@example.com

Sets your email for Git globally (applies to all projects). This information is also used to identify commits.

## 12. git remote add origin GitHubRepositoryHTTPS-link.git

Links your local repository with the corresponding online repository on GitHub.

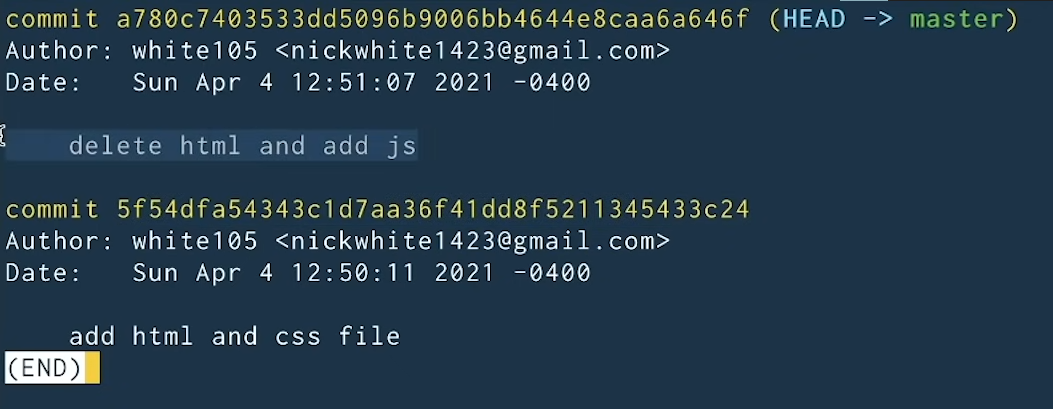
**origin is a name (alias) for the URL of your GitHub repository.**

## 13. git push origin branchName (ex: master)

Pushes the changes in your local repository to the master branch (or any branch) of the online repository. Ensure you have committed your changes before pushing.

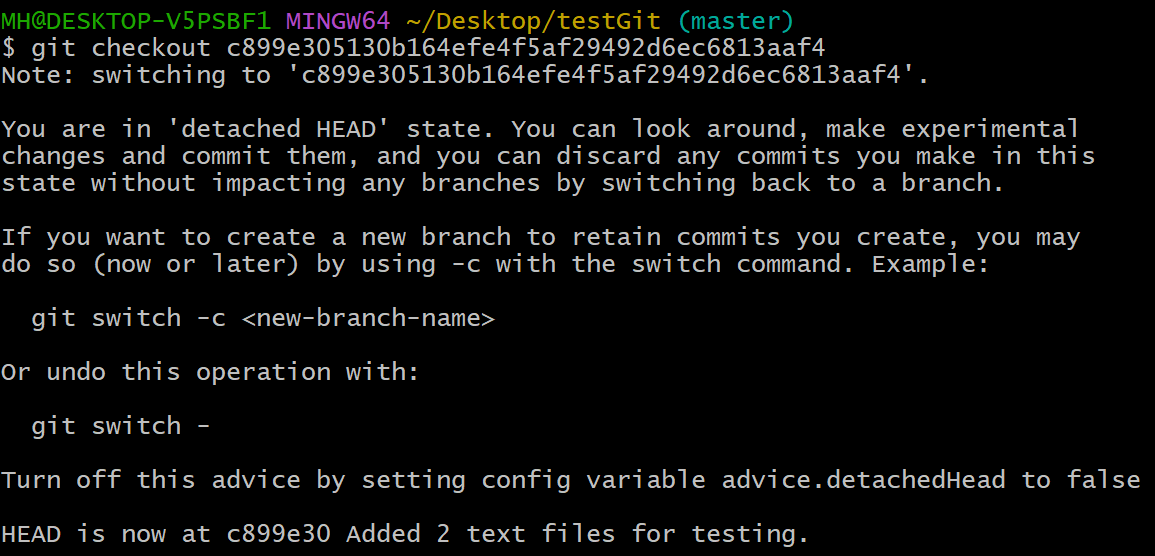
## 14. git log

Displays all changes that have been done on the repository with specific details.



## 15. git checkout commitHashCode

To return to a previous version of commits.



## 16. git checkout -b newBranchName

Creates a new branch

## 17. merging & pull requests of branches

All are done through the GUI on GitHub

## 18. git branch

Displays Branches

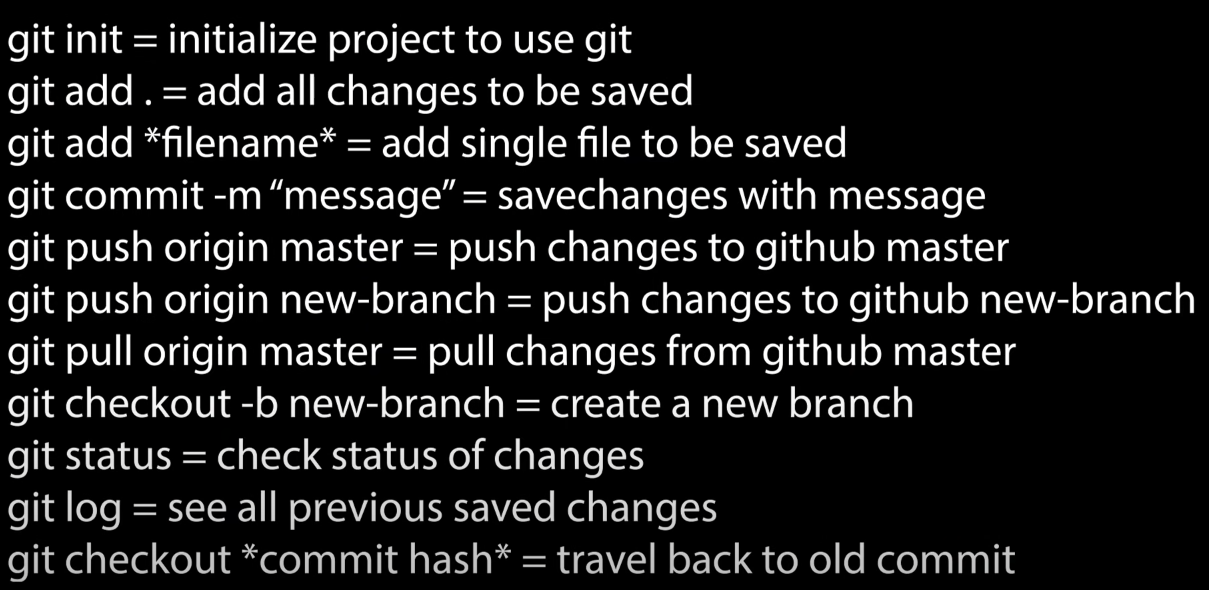
## 18. git switch branchName

Switches to a branch

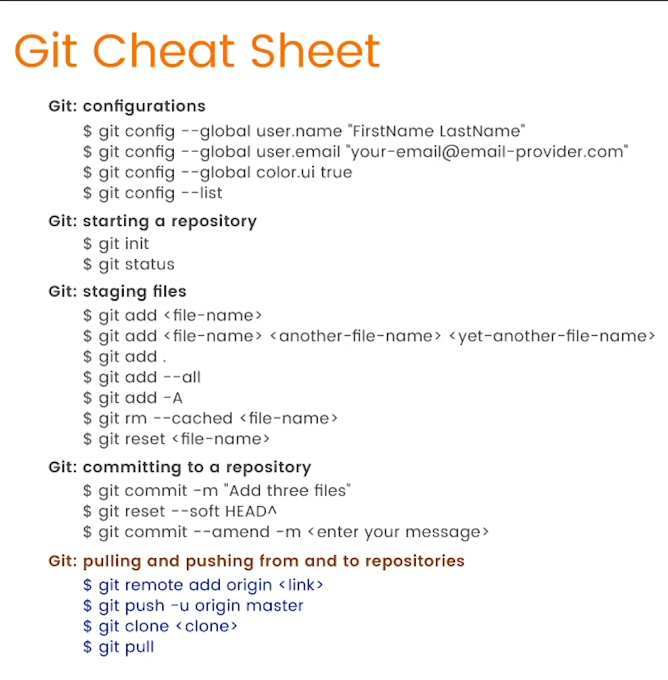
## 19. git pull origin master {NECCESARY BEFORE PUSHING ANYTHING}

Syncs up your local branch to the online branch with all updated changes done there!

## SUMMARY:

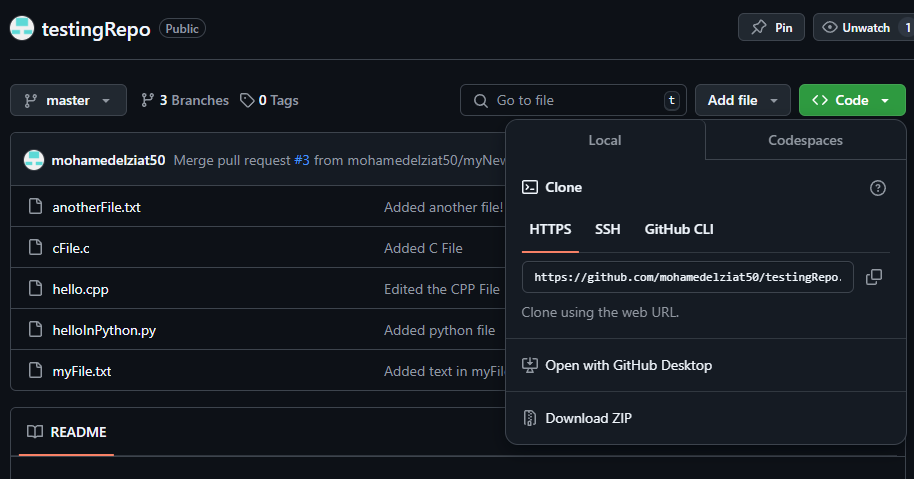


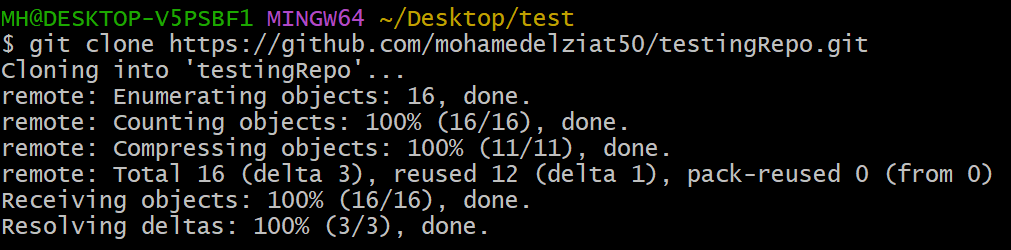
**Next Page -->**



## 21. Steps to Start Working on a GitHub Project Using git clone

1. Access the GitHub Repository:  
 - Obtain the HTTPS link for the repository you want to work on (e.g., from GitHub).

  
  
2. Open Terminal or Git Bash:  
 - Navigate to the folder where you want to clone the repository using the cd command.  
 - Example:  
 cd ~/Desktop  
  
3. Clone the Repository:  
 - Run the git clone command with the HTTPS link:  
 git clone https://github.com/username/repository-name.git  
 - This creates a folder in your current directory containing the repository files.

  
  
4. Navigate into the Cloned Repository:  
 - Use the cd command to move into the project folder:  
 cd repository-name  
  
5. Start Working on the Files:  
 - Make your changes to the project files using your preferred editor or IDE.  
  
6. Synchronize Changes:  
 - After making edits, you can:  
 - Use git add to stage the changes.  
 - Commit the changes using git commit -m 'message'.  
 - Push the changes using git push origin branch-name.