When using a terminal, the following commands are super important.

Commands are in <example>, while arguments for those commands are in [example], and if an argument takes any parameters, they will be in “example”.

**Important Git Commands**

<git config [-global] [user.name “name”] [user.email “email”]>

This command is generally initially used for setting ones name and email

<git init [“repository name”]>

This command creates an empty git repository.

<git clone [“url”]>

This command copies a Git Repo using ssh or https, depending on if you are using a Linux console or a Git Bash terminal.

<git add [“file(s)”] [\*] [-A]>

This command adds a file (or files) to your commit, and \* or -A adds all unadded files with changes not in the ignore list.

<git commit [-m “commit message”]>

This command commits your changes to your repository. You can add a commit message in the command or your terminal will automatically open up an editor for you to do so.

<git diff [-staged] [“first branch” “second branch”]>

This command shows the difference between the requested files or folders.

<git reset [“file”] [“commit”] [-hard “commit”>

This command undoes a commit of a file while preserving the changed file contents. However, using -hard will drop all the contents and set you back to where you were at the point you requested.

<git branch [“branch name”] [-d “branch name”]>

By itself, this command lists all branches. When using just a new name, it creates a branch, and with -d, destroys a specified branch.

<git checkout [“branch name”]

This command switches to the targeted branch.

<git merge [“branch name”]>

This command merges the specified branch back into main.

<git pull>

This command gets all new changes not stored locally from the repository.

<git push>

This command pushes any commits to the repository.

These are the bare minimum commands. For the full list of 20 commands with pictures, please reference:

<https://dzone.com/articles/top-20-git-commands-with-examples>