**$ git init [project-name]**

Creates a new local repository with the specified argument

**$ git status**

Lists all new or modified files to be committed

**$ git config --global user.name "[user-name]“**

Defines the name you want associated with your commit transactions

**$ git config --global user.email "[user-email-address]"**

Defines the email address you want associated with your commit transactions

**$ git config --global color.ui auto**

Turns on colorization of command line output

**$ git add [file]**

Prepares the file for commit by logically moving it to the staged area

**$ git ls-files --stage**

Lists all the files in the staged area

**$ git commit -m "[commit message]"**

Adds the staged files permanently in version history

**$ git diff**

Shows unstaged file differences

**$ git diff --staged**

Shows file differences between staging and the last file version

**$ git rm <file name>**

Remove file from local repo and work folder

**$ git rm --cached <filename>**

Remove file from staging

**$ git branch**

Lists all branches in the current local repository

**$ git branch [branch-name]**

Creates a new branch

**$ git checkout [branch-name]**

Switches to the specified branch and updates the working directory

**$ git merge [branch-name]**

Combines the specified branch’s history into the current branch

**$ git branch -d [branch-name]**

Deletes the specified branch

**$ git rm [file]**

Deletes the file from the working directory and the staging area

**$ git rm --cached [file]**

Removes the file from version control but retains the file locally

**$ git log**

Lists version history for the current branch

**$ git log --oneline**

Lists version history in one line for the current branch

**$ git log --oneline --decorate --graph**

Lists version history in one line, decorated in graphical form for the current branch

**$ git push [alias] [branch]**

Uploads all local branch commits to remote repository

**$ git pull**

Downloads from remote repository and incorporates changes

**$ git stash**

Temporarily stores all modified tracked files

**$ git clone [repository-url]**

Clones an existing repository

**$ git rebase [branch]**

Rebases your current HEAD onto [branch]