Command	Description	Examples (command line)	Result
`git init`	Create a new git repository in the current local directory.	$ mkdir some-project && cd some-project $ git init	This creates a new project directory and then initializes an empty git repository in it. Now when ever you add or change files here git will know about it.
`git clone`	Clones (copies) an existing repository from some remote location like GitHub	$ git clone https://github.com/expressjs/express.git $ git clone https://github.com/expressjs/express.git my-proj	Clones repo in a new dir. called express Clones repo in a new dir. called my-proj
`git status`	Displays the status of all files tracked by git (changed, deleted, moved, or new/ untracked files)	$ git status	Shows the files that have changed, broken up into those staged for commit and those with unstated changes.

`git add`	Converts the specified unstated files into staged files	$ git add .
		$ git add index.js
		$git add **/*.js
			Adds all files in the project to staged
			Adds the “index.js” file to staged
			Adds all the javascript files in the project to staged.

`git checkout`	Removes files that are already tracked from staged (changes will be reverted).	$ git checkout index.js	Removes the file “index.js” from staged/ unstaged and reverts changes to the previous commit.
`git rm`	Removes files that are untracked, using the —cached flag will allow the file to continue to exist, it just won’t be tracked by git.	$ git rm index.js $ git rm —cached index.js	Deletes the file “index.js” Removes the untracked “index.js” file from “untracked” but does not actually delete the file. This lets you decide what to do with the file in a future commit.
`git commit`	Creates a new commit on the current branch	$ git commit (next you can review the diff and write a commit message)	After saving your message a new commit is created in the current branch, this commit can be compared to other commits to see the difference, or checked out to take your directory back to this point in time
`git log`	Shows a chronological history of all the commits in the current branch	$ git log	Shows a chronological history of all the commits in the current branch, review the log and press *q* to exit.
`git pull`	Pulls down changes from a remote host to your local copy	$ git pull origin master	Pulls the latest commits you don’t have from origin’s (probably GitHub) master branch into your current local branch. In this example you’d probably want to be on Master branch when you do this.

Command	Description	Examples (command line)	Result
`git push`	Pushes your local changes in the current branch to the specified remote branch	$ git push origin master	Pushes the latest commits fro your current branch to origin's Master. Again you'd probably want to be on local Master when you do this, although, when working with a team you'll like likely be working in and pushing to a branch other than master.

`git branch`	Branch is a command for working with branches (lists, create, delete, etc.	$ git branch
		$ git branch -a
		$ git branch -d my-branch
			Lists all the local branches
			Lists all the local and remote branches
			Deletes the specified branch

`git fetch`	Pulls down a list of all the current branches from the remote repository	$ git fetch origin	Adds to your local branches references to all the new remote branches you don't currently have
`git checkout branch-name`	Git checkout again, if you specify a branch name instead of a file name git will checkout that branch, the **-b flag** will create a new branch	$ git checkout develop $ git checkout -b my-branch	Checkouts out the develop branch Creates and then checkout a new branch called "my-branch". When changing branches all of the files in your project directory automatically change to the way they are in that branch.