Git

Git is a piece of software that makes it simpler to manage different versions of your own software in a software repository.

GitHub

GitHub is a place to share software repositories and collaborate with other developers. Github is a website where you host all of your Git repositories.

These tools make it easier for multiple people to work on the same software project.

To create bullet points in Markdown on GitHub, start each line with an asterisk (*), hyphen (-), or plus sign (+), followed by a space, and then the text for the bullet point.

In this course you will learn about:

- common terms
- installing Git
- creating a Git repository
- cloning a Git repository
- adding and removing from a Git repository
- using GitHub
- creating pull requests
- branching repositories
- merging repositories
- and more

What is Version Control?

What is Git?

A: Free and open source version control system.

What is Version Control?

A: The management of changes to documents, computer programs, large web sites, and other collections of information.

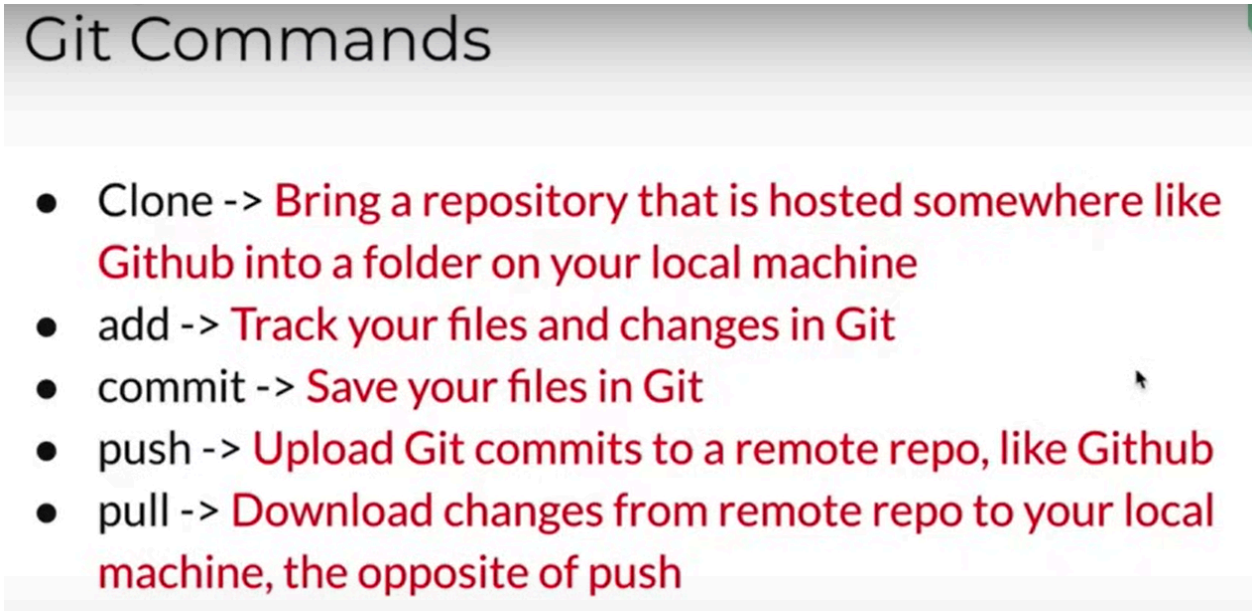
VC is basically a way **programmers track their code changes.** They can look back at **all of their code changes** they have **made over time.** It helped them

- **to see and understand what they did**
- **Track down bugs**
- **Go back to previous versions of the code if they need to**

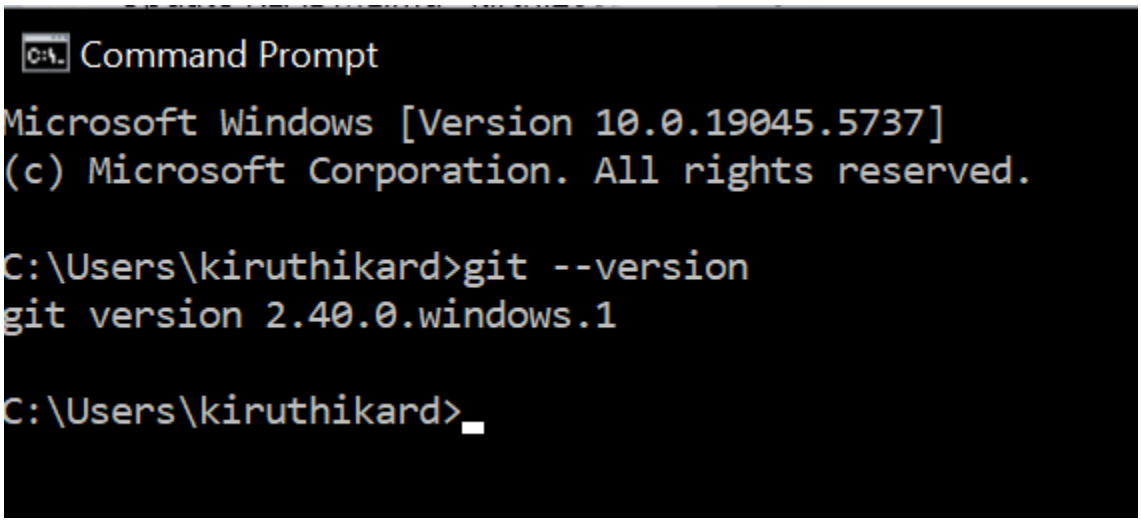
Terms

- Directory -> Folder
- Terminal or Command Line -> Interface for Text Commands
- CLI -> Command Line Interface
- cd -> Change Directory
- Code Editor -> Word Processor for Writing Code
- Repository -> Project, or the folder/place where your project is kept
- Github -> A website to host your repositories online

Git Commands



How to find Git version in command prompt?



Link to install Git

https://www.atlassian.com/git/tutorials/install-git

Link to install Visual Studio

https://www.code.visualstudio.com

Git Clone

git clone https://github.com/kirthi2005/Git_GitHub_Freecodecamp_Course.git - Clone the remote repository local to your machine

Git Status

git status - command shows all the files that were created or updated or deleted but not saved in commit yet.

Git Add

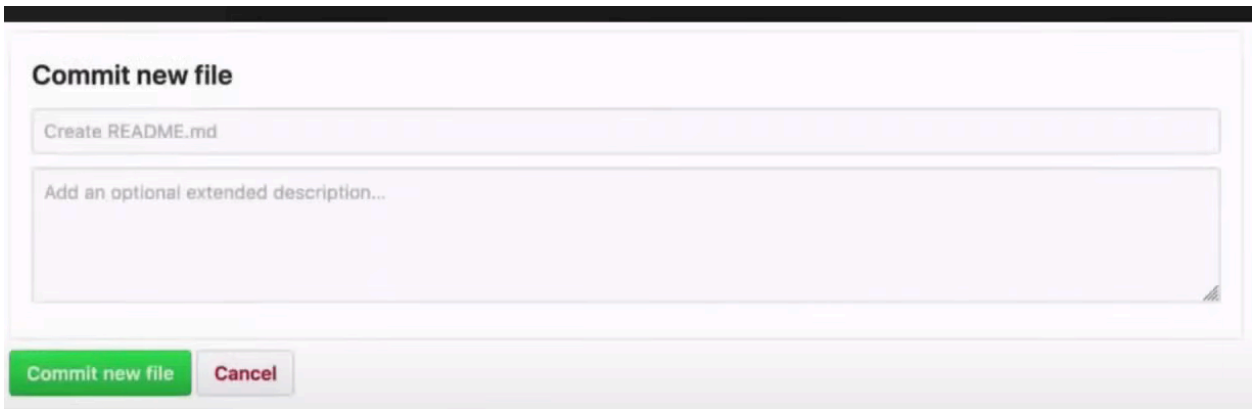
git add . -> command to tell git to track all of the files the modified and the intracked files (new files) (stage the changes)

git add filename - to add specific file

Git Commit

git commit -m "Added Readme" -m "you can add description here"

-m is for message



Git Push

git push origin master or git push origin main - push our changes in local machine to remote repository. Origin - location of our github repository.master/main are the branch where we want to put our code

SSH Keys

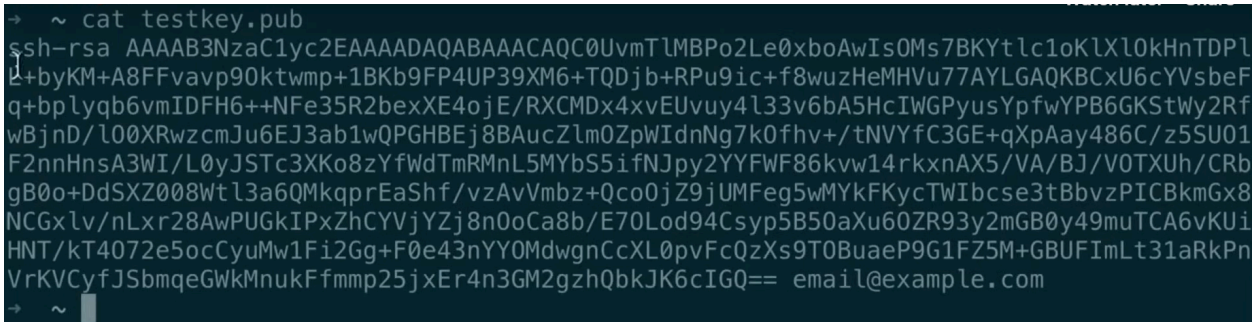
Used to connect your local machine and remote repository safely. These keys are used to identify you are owner of the repository.

```
➜  ~ ssh-keygen -t rsa -b 4096 -C "email@example.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/gwen/.ssh/id_rsa): testkey
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in testkey.
Your public key has been saved in testkey.pub.
The key fingerprint is:
SHA256:gHmbQn8S/hAhl2GMFzAap1moz2HYzD0wf6LkisCW9J0 email@example.com
The key's randomart image is:
+---[RSA 4096]----+
|  ..*+*+         |
|  +B.B+.         |
| *+=+.=          |
|o.B.=+.*         |
|o*ooo+O S        |
|.+=. E =         |
|+.      .        |
|o                |
|                 |
+----[SHA256]-----+
➜  ~
```

ssh-keygen - generate ssh key
-t rsa - algorithm name
-b 4096 strength of the key
Email id

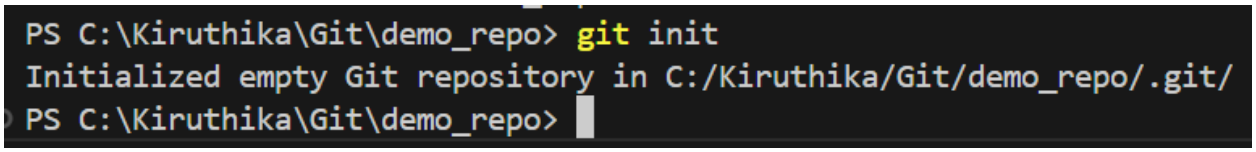
```
➜  ~ ls | grep testkey
testkey
testkey.pub
➜  ~
```

testkey is a private key. You have keep secured in your local machine
testkey.pub - indicates the key is available to the public. This is the one we are going to upload in github
The public will look like below



Git Init

git init - initialize git repository in the folder in your local machine



Git Remote

Create remote repository first and then push our local code.

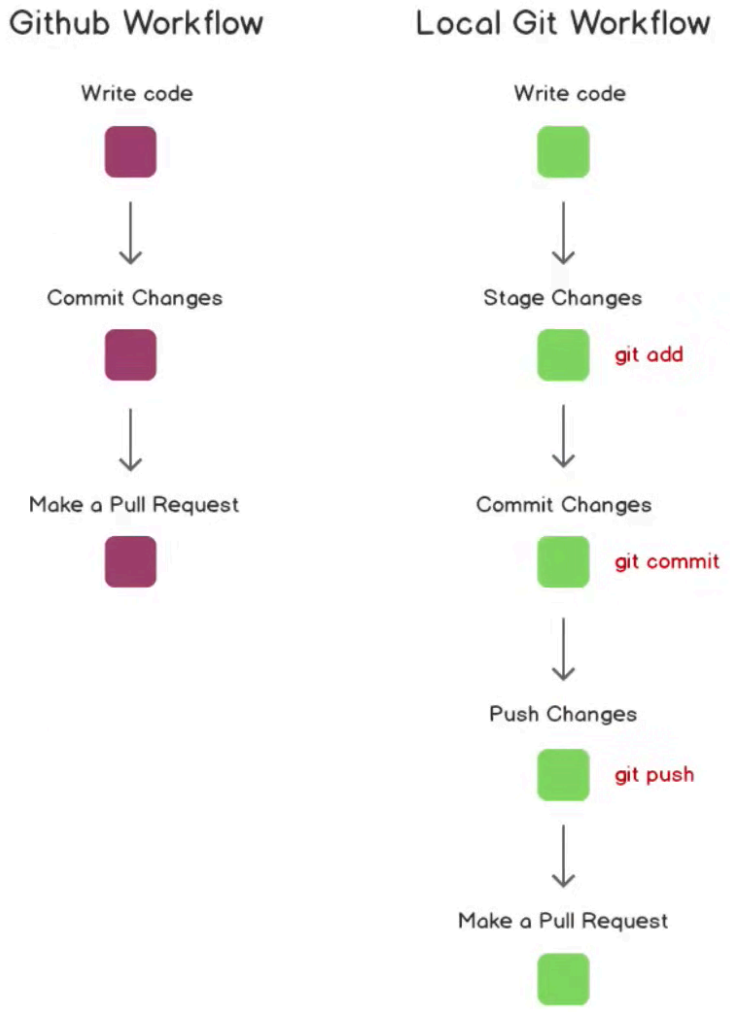
git remote add origin "your remote repository url" - add reference to the remote repository on github.

git remote -v - show any remote repositories that I've connected to this repo.

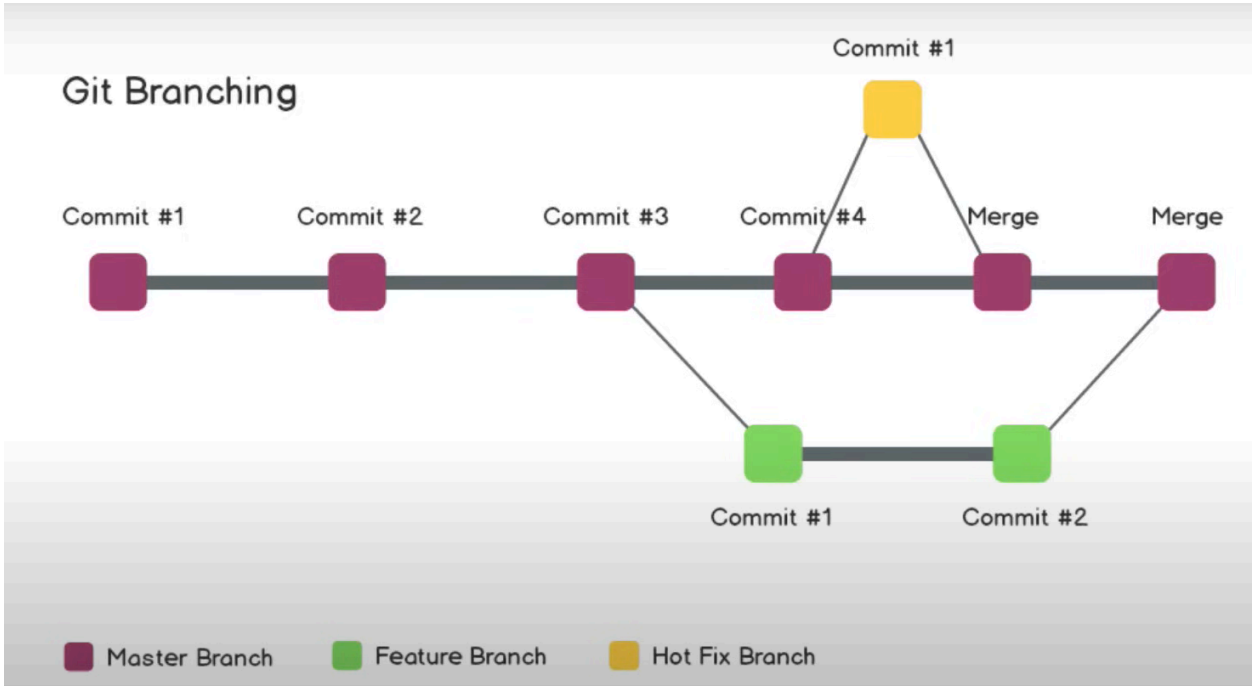
git push origin master

git push -u origin master (-u is upstream. No need to type origin master everytime)

Git and Local workflow



Git Branching



Code in master branch and feature branch will be the same. But if you do changes in feature branch it won't get reflected in master branch.

```
→  demo-repo git:(master) git branch
→  demo-repo git:(master) git checkout -b feature-readme-instructions
Switched to a new branch 'feature-readme-instructions'
→  demo-repo git:(feature-readme-instructions) git branch
→  demo-repo git:(feature-readme-instructions) git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
→  demo-repo git:(master) git branch
→  demo-repo git:(master) git checkout feature-readme-instructions
```

git checkout -b branchname -> -b will create new branch
git checkout master - switch to master branch
git checkout feature - switch to feature branch

```
→  demo-repo git:(master) git checkout feature-readme-instructions
Switched to branch 'feature-readme-instructions'
→  demo-repo git:(feature-readme-instructions) git status
On branch feature-readme-instructions
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
→  demo-repo git:(feature-readme-instructions) ✗
```

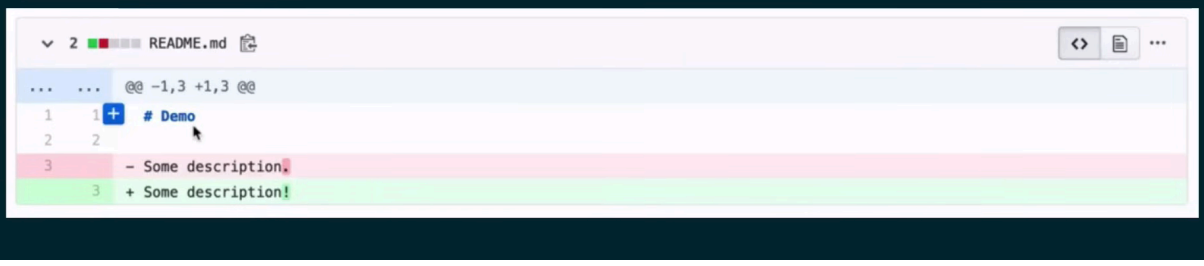
MORE VIDEOS

```
→  demo-repo git:(feature-readme-instructions) ✗ git add README.md
→  demo-repo git:(feature-readme-instructions) ✗ git commit -m "updated readme"
[feature-readme-instructions cec9aeb] updated readme
 1 file changed, 4 insertions(+)
→  demo-repo git:(feature-readme-instructions) git checkout mast
```

```
 1 file changed, 4 insertions(+)
→  demo-repo git:(feature-readme-instructions) git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
→  demo-repo git:(master)
```

Git diif

```
→  demo-repo git:(master) git diff
```



git diff feature-readme-instructions

```
diff --git a/README.md b/README.md
index ac186b2..2b23fdd 100644
--- a/README.md
+++ b/README.md
@@ -5,7 +5,3 @@ Some description!
 ## Subheader

 Watch tutorial on YouTube.
-
-## Local Development
-
-1. Open index.html in your browser.
(END)
```

Git Merge

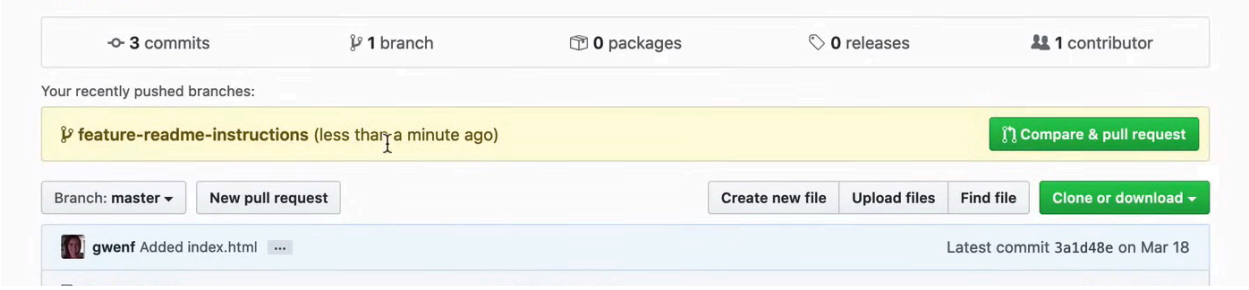
```
➜  demo-repo git:(master) git diff feature-readme-instructions
➜  demo-repo git:(master) git merge feature-readme-instructions
```

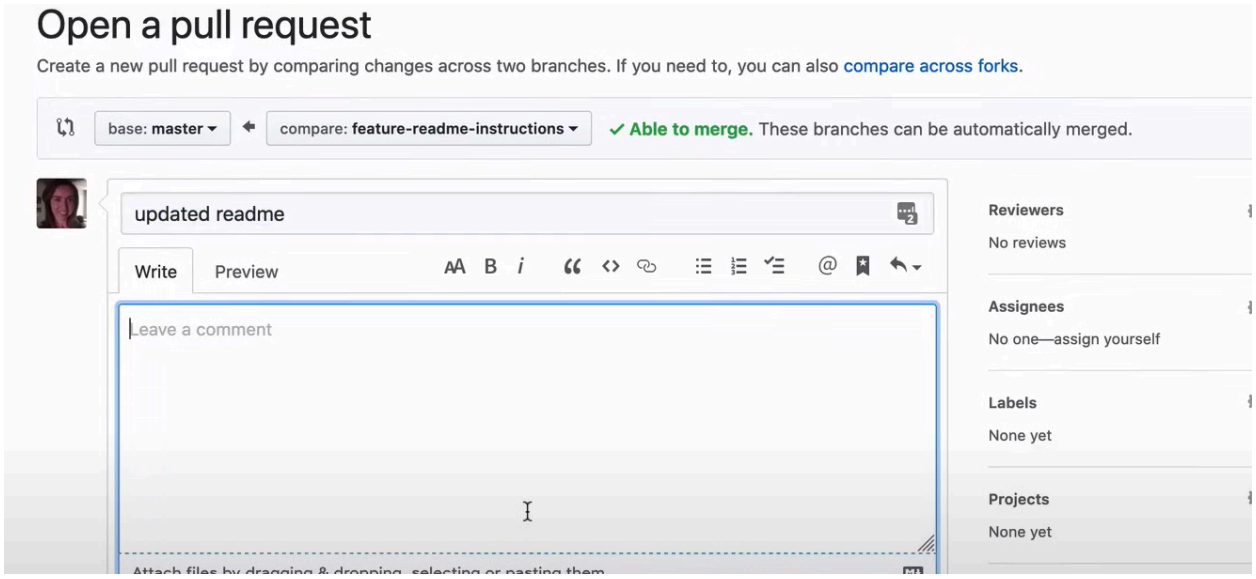
```
➜  demo-repo git:(master) git diff feature-readme-instructions
➜  demo-repo git:(master) git checkout feature-readme-instructions
Switched to branch 'feature-readme-instructions'
➜  demo-repo git:(feature-readme-instructions) git status
On branch feature-readme-instructions
nothing to commit, working tree clean
➜  demo-repo git:(feature-readme-instructions) git push
fatal: The current branch feature-readme-instructions has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin feature-readme-instructions

➜  demo-repo git:(feature-readme-instructions)
```

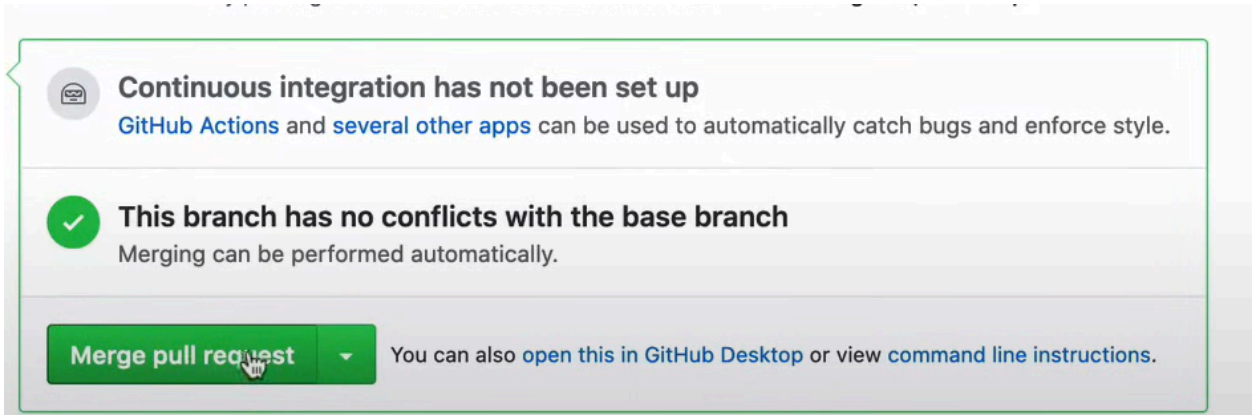
```
→  demo-repo git:(feature-readme-instructions) git push -u origin feature-readme-ins
tructions
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 380 bytes | 380.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'feature-readme-instructions' on GitHub by visitin
g:
remote:      https://github.com/gwenf/demo-repo/pull/new/feature-readme-instructions
remote:
To github.com:gwenf/demo-repo.git
 * [new branch]      feature-readme-instructions -> feature-readme-instructions
Branch 'feature-readme-instructions' set up to track remote branch 'feature-readme-i
nstructions' from 'origin'.
→  demo-repo git:(feature-readme-instructions)
```

Pull request

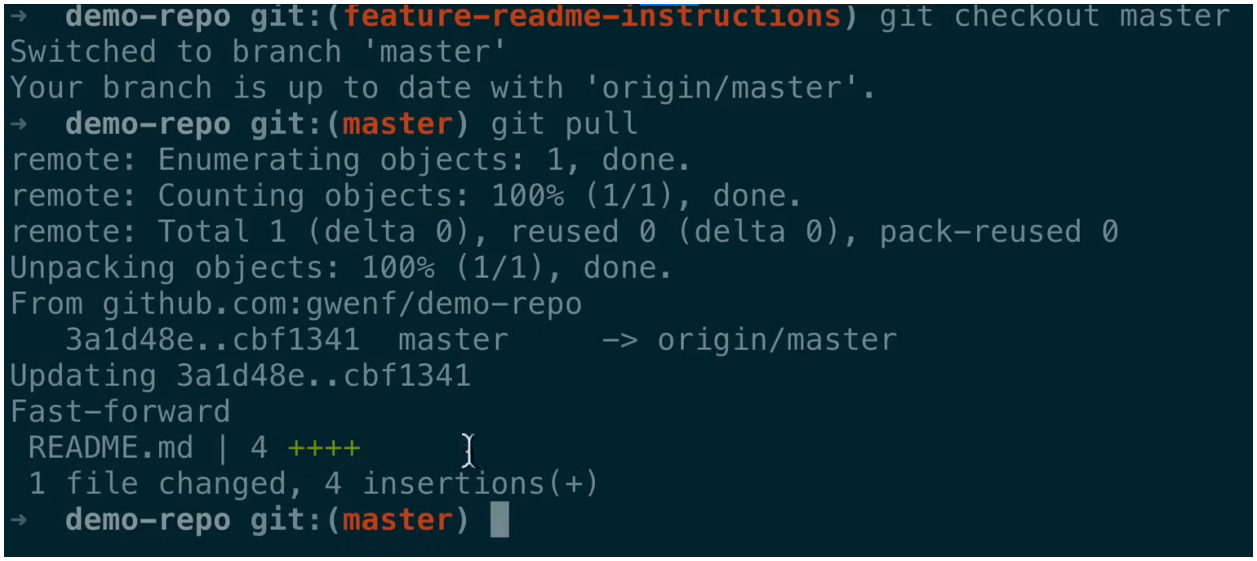
It is basically a to have your code pulled into another branch. In the example, we are making PR from the feature branch to the master branch.once we have made a PR anyone can review our code ,comment on it, ask us to make changes or updates.After you make a PR you can also update the code just by making additional commits and pushing them upto Github, as long as u r on the branch where u made the PR.Once PR is merged delete your feature branch and switch back to master branch.







Pull code to local master branch



Git branch delete

```
➜  demo-repo git:(master) git branch
➜  demo-repo git:(master) git branch -d feature-readme-instructions
Deleted branch feature-readme-instructions (was cec9aeb).
➜  demo-repo git:(master)
```

Merge Conflicts

Different people working on the same file and merging their own code at same time in master branch. So sometime Git doesn't know which code to keep or which code is redundant or which code you want to get rid of.

Commit and add at the same time

It works only for modified files.

git commit -am "commit message"

Undoing in Git

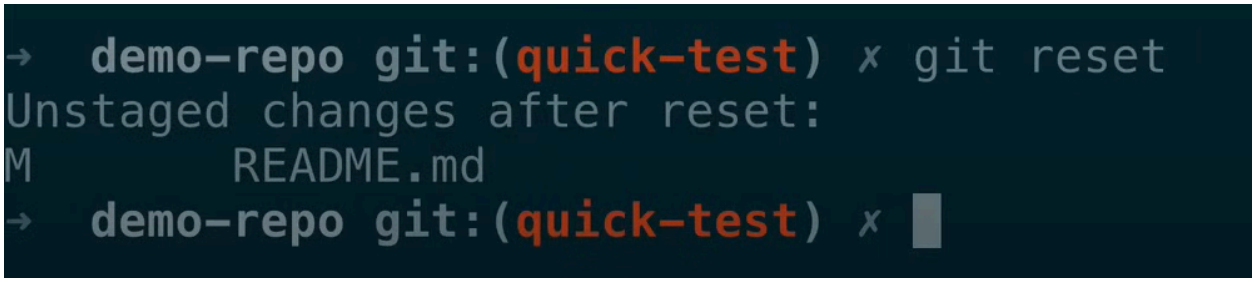
Changes are staged

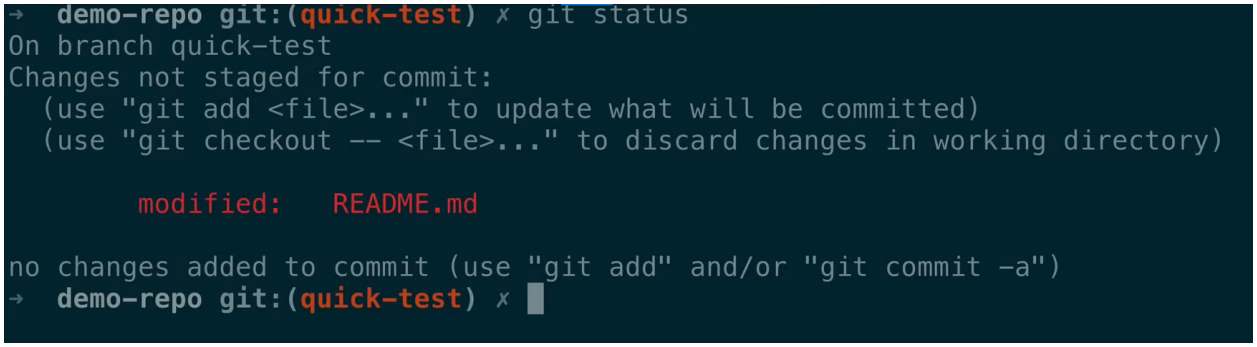
```
➜  demo-repo git:(quick-test) git status
On branch quick-test
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
➜  demo-repo git:(quick-test) ✗ git add README.md
➜  demo-repo git:(quick-test) ✗ git status
On branch quick-test
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md

➜  demo-repo git:(quick-test) ✗
```

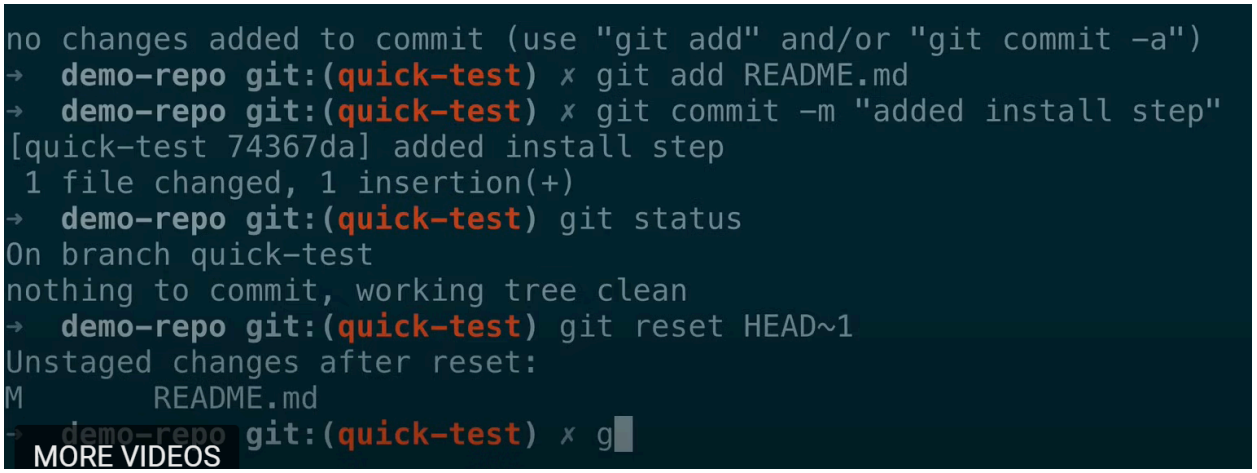
Undo the changes

git reset or git reset filename (name of the file to unstage)

git reset --hard id# (not only instaged but completely removed)





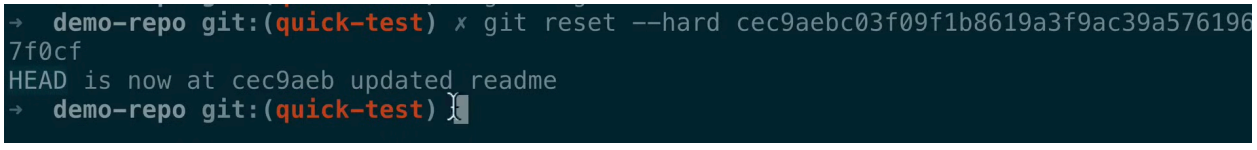


HEAD points to last commit and ~1 is go back before one step of head

Git log

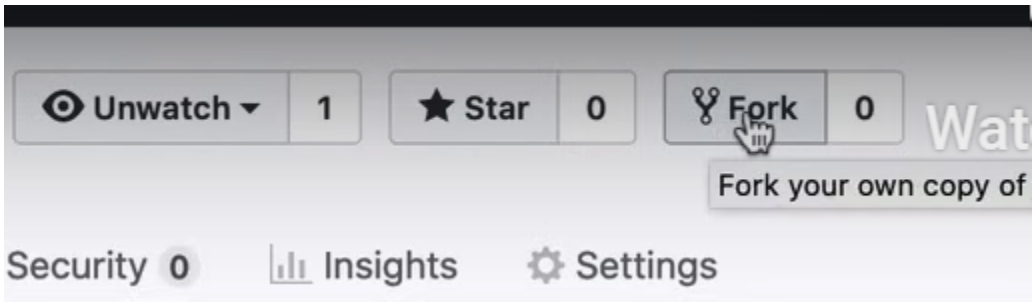
git log - see a log of all your commits. Logs will be arranged in reverse chronological order.

Git Reset Hard



Fork

To make a complete copy of the repository



YouTube Video Link

https://www.freecodecamp.org/news/git-and-github-crash-course/

GitBash Command Review

Git bash basic commands review : https://www.youtube.com/watch?v=oQc-2gsjgDg

Top 20 Git Commands

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

How to understand Git: an intro to basic commands, tips, and tricks

https://www.freecodecamp.org/news/understanding-git-basics-commands-tips-tricks/