Git Documentation

A Git project will have the following three main sections:

1. **Git directory** - local file where git stores everything it needs commonly found in `YOUR-PROJECT-PATH/.git/`
2. **Working directory** is where a user makes local changes to a project. The working directory pulls the project's files from the Git directory's object database and places them on the user's local machine.
3. **Staging area** that stores information about what will go into your next commit. A commit is when you tell Git to save these staged changes. Git takes a snapshot of the files as they are and permanently stores that snapshot in the Git directory.

Install Git

- Ubuntu: `sudo apt-get install git`
- Windows: Download
- Mac: Download

Configure the Git Environment

Add Your Name and Email

Git includes the user name and email as part of the information in a commit. You'll want to set this up under your user-level configuration file with these commands:

```
git config --global user.name "My Name"
```

```
git config --global user.email myemail@example.com
```

```
APPLEs-MacBook-Pro:~ vishalvaitheeswaranrk$ git config --global user.name "VishalVrk"
APPLEs-MacBook-Pro:~ vishalvaitheeswaranrk$ git config --global user.email "vishalvrk97@gmail.com"
APPLEs-MacBook-Pro:~ vishalvaitheeswaranrk$ clear
APPLEs-MacBook-Pro:~ vishalvaitheeswaranrk$
```

Initialize Git in a Project

This installs a Git directory folder with all the files and objects Git needs to track your project. cd to your project folder and enter the command

```
git init
```

```
APPLEs-MacBook-Pro:desktop vishalvaitheeswaranrk$ cd git\ documentation
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ clear
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git init
Initialized empty Git repository in /Users/vishalvaitheeswaranrk/Desktop/git documentation/.git/
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$
```

You can see that you have a working directory in the local machine

Checking the Status of Your Files

If you run this command directly after a clone, you should see something like this:

$ git status

On branch master Your branch is up-to-date with 'origin/master'.

nothing to commit, working directory clean

Let's say you add a new file to your project, a simple README file. If the file didn't exist before, and you run git status, you see your untracked file like so:

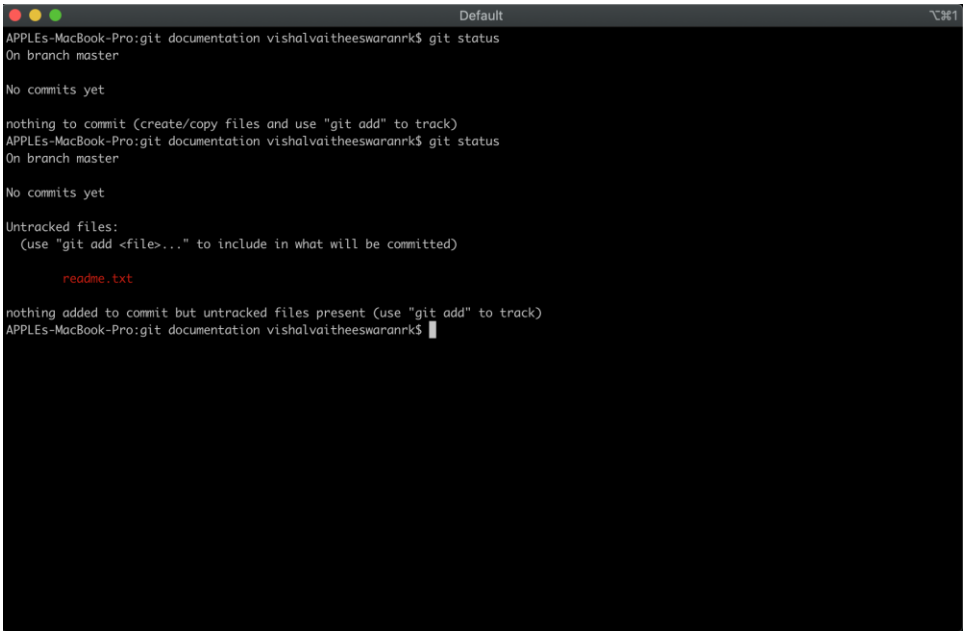
$echo 'My Project' > README

$ git status

On branch masterYour branch is up-to-date with 'origin/master'.

Untracked files: (use "git add <file>..." to include in what will be committed)

README nothing added to commit but untracked files present (use "git add" to track)



Tracking New Files

In order to begin tracking a new file, you use the command git add. To begin tracking the README file, you can run this:

$ git add README

git add <files>

To add all the untracked files in to stage you can use the command

git add .

```
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git add readme.txt
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   readme.txt

APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ clear
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$
```

Ignoring Files

Often, you'll have a class of files that you don't want Git to automatically add or even show you as being untracked. These are generally automatically generated files such as log files or files produced by your build system. In such cases, you can create a file listing patterns to match them named .gitignore. Here is an example .gitignore file:

Here you can see that files with *.ignore will not be staged

Committing Your Changes

git commit

Doing so launches your editor of choice. For eg vim

When you exit the editor, Git creates your commit with that commit message (with the comments and diff stripped out).

```
Default                                                    ⌥⌘1
Last login: Sat Apr  4 17:00:37 on ttys000
APPLEs-MacBook-Pro:~ vishalvaitheeswaranrk$ cd ddesktop
-bash: cd: ddesktop: No such file or directory
APPLEs-MacBook-Pro:~ vishalvaitheeswaranrk$ cd desktop
APPLEs-MacBook-Pro:desktop vishalvaitheeswaranrk$ cd git\ documentation
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git commit
[master (root-commit) 7d10ae4] this is first commit :x
 2 files changed, 2 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 readme.txt
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git log
commit 7d10ae41867168d3cd2e90be56545b2c5eb92aea (HEAD -> master)
Author: VishalVrk <vishalvrk97@gmail.com>
Date:   Sat Apr 4 17:22:48 2020 +0530

    this is first commit
    :x
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$
```

Alternatively, you can type your commit message inline with the commit command by specifying it after a -m

git commit -m "Story 182: fix benchmarks for speed"

```
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git status
On branch master
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:   .gitignore

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        readme2.txt

no changes added to commit (use "git add" and/or "git commit -a")
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git add readme2.txt
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git commit -m 'Added
 readme2.txt newly'
[master 0f662a5] Added readme2.txt newly
 1 file changed, 1 insertion(+)
 create mode 100644 readme2.txt
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$
```

Adding Remote Repository

git remote add origin <repo url>

```
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ clear
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git remote add origin https://github.com/VishalVrk/git-documentation.git
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$
```

Pushing staged files to remote repository

git push -u origin master

```
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git push -u origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 461 bytes | 461.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/VishalVrk/git-documentation.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Basic Branching

To create a new branch and switch to it at the same time, you can run the git checkout command with the -b switch:

$ git checkout -b iss53

Switched to a new branch "iss53"

```
Default
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git checkout -b newBranch
M       .gitignore
Switched to a new branch 'newBranch'
```

Basic Merging

Suppose you've decided that your issue #53 work is complete and ready to be merged into your master branch.

$ git checkout master

Switched to branch 'master'

$ git merge iss53

Merge made by the 'recursive' strategy.

```
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git checkout master
M       .gitignore
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
APPLEs-MacBook-Pro:git documentation vishalvaitheeswaranrk$ git merge newBranch
Updating 0f662a5..dd61cfc
Fast-forward
 readme3.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 readme3.txt
```

Create a new Pull Request

Consider a scenario where you're a team and you work on several tasks, you can create a separate branch for your workflows on you remote repository and after committing your changes to the branch you can create a pull request to validate your changes and to make sure that it doesn't change master branch.

Step 1: checkout new branch

Step 2: Commit and push to remote repository

Step 3: Create a pull request on comparing with master branch / Main Branch

Update readme2.txt #4

Edit

Open VishalVrk wants to merge 1 commit into `master` from `VishalVrk-changed-readme2`

Conversation 0 | Commits 1 | Checks 0 | Files changed 1 | +2 −1

VishalVrk commented 39 seconds ago — Owner

No description provided.

`Update readme2.txt` Verified a6f5e48

Add more commits by pushing to the **`VishalVrk-changed-readme2`** branch on **VishalVrk/git-documentation**.

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request — You can also open this in GitHub Desktop or view command line instructions.

Write | Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Linked issues
Successfully merging this pull request may close these issues.
None yet

Notifications Customize
Unsubscribe