

29

Git Version Control System (VCS)

Basics:

Install:

```
# apt install git -y
# apt install gh -y
# git --version
```

When you start using Git, you'll need to put in username and email id:

```
# git config --global user.name "username"
# git config --global --list
```

OR

```
git config --global -l
```

Git Clone:

```
# git clone https://github.com/account_name/repo.git
# cd repo/
# ls -a
. git
```

To initializing a git repository in the current directory (specified in your terminal).

```
# mkdir dir
# cd dir
# git init
```

```
# ls -a
. git/
```

```
# git add
```

- Create a branch :
- ```
git checkout -b first
```
- b: branch name.

used to switch and create a new branch.

- Switch to branch :
- ```
# git checkout first
```

- List branch :
- ```
git branch.
```

- List Remove Branches
- ```
# git branch -a.
```

```
# cat >> readme.md
||||
```

```
# git diff
diff --git a/readme.md b/readme.md
index 039727e..11d149e 100644
--- a/readme.md
+++ b/readme.md
@@ -1,2 @@
lol
+||||
```

```
# git add readme.md
OR
```

```
# git add. ← add's all files & folders.
```


Commit

git commit -m "lol1" ← comment.

git status

git log

Merge 'first' branch with 'main' branch:

git checkout main

git merge first

Viewing Your Comments.

git log

One line Histories.

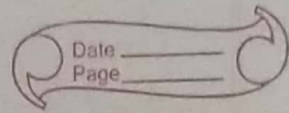
git log --pretty=oneline

Controlling which entries are Displayed.

git log --pretty=oneline --max-count=2
--since='5 minutes ago'
--until='5 minutes ago'
--author=<your name>
--all

git log --pretty=oneline | wc -l ← counts no. of log

#git push -u origin main



Git Internals :

The .git directory

ls -C .git

The Object store

ls -C .git/objects

You should see a bunch of directories with 2 letter names. The directory names are the first two letters of the SHA1 hash of the object stored in git.

Deeper into the Object Store:-

ls -C .git/objects/<dir>

- Look in one of the two-letter directories.
- You should see some files with 38-character names.
- These are the files that contain the objects stored in git.
- These files are compressed and encoded, so looking at their contents directly won't be very helpful, but we will take a closer look in a bit.

Config file:

cat .git/config

- This is a project-specific configuration file.

Branches & Tags :


```
# ls .git/refs
# ls .git/refs/heads
# ls .git/refs/tags
# cat .git/refs/tags/v1
```

Each file corresponds to a tag you created with the `git tag` command earlier. Its content is just the hash of the commit tied to the tag.

The HEAD file:

```
# cat .git/HEAD
```

The HEAD file contains a reference to the current branch.

Dumping the Latest Commit

```
# git hist --all [not working now]
Using the SHA1 hash from the commit listed above ...
```

Manual way to extract blob's content:

```
# git log --pretty=oneline
or
# git log --stat --pretty=oneline
# git show ab35d03c8f4238...
# git cat-file --batch-check --batch-all-objects |
  grep blob
# git cat-file -p HASH
```

```
# git cat-file --batch-check --batch-all-objects |  
grep blob | awk '{print $1}' | while read  
-r hash; do git cat-file -p $hash; done |  
grep "username\|password\|db_user\|db_pass"
```