

Experiment 1

AIM - To Install Git and their commands

Theory -

Git is the most commonly used version control system. Git tracks the changes you make to files, so you have a record of what has been done, and you can revert to specific versions should you ever need to. Git also makes collaboration easier, allowing changes by multiple people to all be merged into one source. Git is software that runs locally. Your files and their history are stored on your computer. You can also use online hosts (such as GitHub or Bitbucket) to store a copy of the files and their revision history. Having a centrally located place where you can upload your changes and download changes from others, enable you to collaborate more easily with other developers. Git can automatically merge the changes, so two people can even work on different parts of the same file and later merge those changes without losing each other's work!

Commands -

1. git --version

You can check your current version of Git by running the git --version command in a terminal (Linux, macOS) or command prompt (Windows).

Output -

```
(base) Priyams-MacBook-Pro:~ priyamvora$ git --version
git version 2.19.0
(base) Priyams-MacBook-Pro:~ priyamvora$
```

2. git init

The git init command creates a new Git repository. It can be used to convert an existing, unversioned project to a Git repository or initialize a new, empty repository.

Output -

```
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git init
Initialized empty Git repository in
/Users/priyamvora/Academics/Djsce/Sem 8/DevOps/Experiment 1/.git/
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$
```

3. git config --global

Experiment 1

You can specify Git configuration settings with the git config command. One of the first things you did was set up your name and email address. Global level configuration is user-specific, meaning it is applied to an operating system user. Global configuration values are stored in a file that is located in a user's home directory.

Output

```

Experiment 1 — -bash — 101x28
[(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git init
Initialized empty Git repository in /Users/priyamvora/Academics/Djsce/Sem 8/DevOps/Experiment 1/.git/
[(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git config --global
usage: git config [<options>]

Config file location
  --global          use global config file
  --system          use system config file
  --local           use repository config file
  -f, --file <file> use given config file
  --blob <blob-id>  read config from given blob object

Action
  --get             get value: name [value-regex]
  --get-all        get all values: key [value-regex]
  --get-regexp      get values for regexp: name-regex [value-regex]
  --get-urlmatch    get value specific for the URL: section[.var] URL
  --replace-all    replace all matching variables: name value [value_regex]
  --add            add a new variable: name value
  --unset          remove a variable: name [value-regex]
  --unset-all     remove all matches: name [value-regex]
  --rename-section  rename section: old-name new-name
  --remove-section remove a section: name
  -l, --list       list all
  -e, --edit       open an editor
  --get-color      find the color configured: slot [default]
  --get-colorbool  find the color setting: slot [stdout-is-tty]
```

4. git config --global --list
List username and email of users currently using the version control system

Output

```

(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git config
--global --list
user.name=priyamvora
user.email=priyamvora99@gmail.com
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$
```

Experiment 1

5. mkdir 'name'

Used for creating empty directory. Create a directory and change into the directory using cd command. cd directory_name

6. Git status

The git status command displays the state of the working directory and the staging area.

Output

```
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git status
On branch master
No commits yet
nothing to commit (create/copy files and use "git add" to track)
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$
```

7. Git add

The git add command adds a change in the working directory to the staging area.

Output

```
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ touch sample.txt
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git add sample.txt
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git status
On branch master

No commits yet

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

        new file:   sample.txt

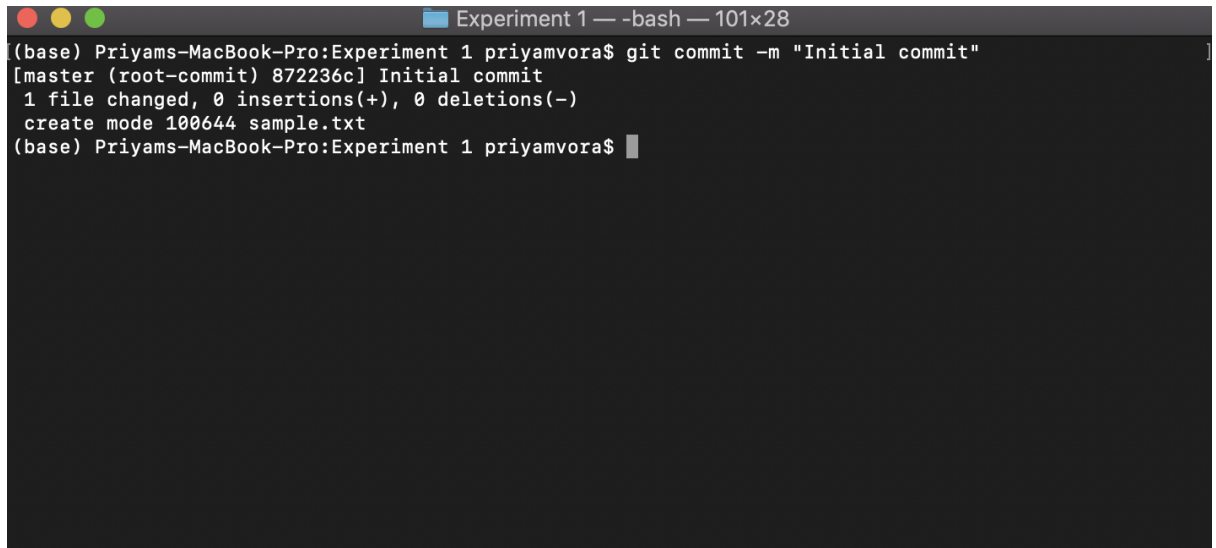
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$
```

8. Git commit

In Git, commit is the term used for saving changes. Git does not add changes to a commit automatically. You need to indicate which file and changes need to be saved before running the Git commit command.

Output

Experiment 1



```

Experiment 1 — -bash — 101x28
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git commit -m "Initial commit"
[master (root-commit) 872236c] Initial commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 sample.txt
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$

```

9. Git log

Git log is a utility tool to review and read a history of everything that happens to a repository

Output

```

(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git log
commit 872236c57a159ed728aab60a737c88b7e6095dcb (HEAD ->
master)
Author: priyamvora <priyamvora99@gmail.com>
Date: Thu Feb 25 13:57:31 2021 +0530

```

Initial commit

```

(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$

```

10. Git branch

Git branches are effectively a pointer to a snapshot of your changes. When you want to add a new feature or fix a bug—no matter how big or how small—you spawn a new branch to encapsulate your changes. This makes it harder for unstable code to get merged into the main code base, and it gives you the chance to clean up your future's history before merging it into the main branch.

Output

```

(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git branch
* master

```

```

(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git branch branch1
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git branch

```

Experiment 1

branch1

* master

```
[(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git branch ]
* master
[(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git branch branch1 ]
[(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ git branch ]
branch1
* master
(base) Priyams-MacBook-Pro:Experiment 1 priyamvora$ █
```