Subject : Source Code Management (SCM)

Subject Code : CS181

Cluster : Beta

Department : CSE

SUBMITTED BY:

Name : Diksha
Roll No. 2110990446
Group : G08

SUBMITTED TO:

Dr. Monit Kapoor

INDEX:

CHITKARA UNIVERSITY

Experiment No. 1

Aim: Setting Up the Os git Clint.

Why git around ?

Git is a version control system. It helps us to

- Easily recover files
- Roll back to previously working state

Installation:

Generally git already exists in Mac OS.
In case it's not, go to -
http://git-scm.com/download/mac to download git.

After visiting the above link y'll be having a page like this:

Download for macOS

There are several options for installing Git on macOS. Note that any non-source distributions are provided by third parties, and may not be up to date with the latest source release.

Homebrew

Install homebrew if you don't already have it, then:

```
$ brew install git
```

MacPorts

Install MacPorts if you don't already have it, then:

```
$ sudo port install git
```

Xcode

Apple ships a binary package of Git with Xcode.

Binary installer

Tim Harper provides an installer for Git. The latest version is 2.33.0, which was released 7 months ago, on 2021-08-30.

Building from Source

If you prefer to build from source, you can find tarballs on kernel.org. The latest version is 2.35.1.

Installing git-gui

If you would like to install git-gui and gitk, git's commit GUI and interactive history browser, you can do so using homebrew

```
$ brew install git-gui
```

It will be easy to install git from Binary installer. Click on installer and go ahead.

Binary installer

Tim Harper provides an installer for Git. The latest version is 2.33.0, which was released 7 months ago, on 2021-08-30.

Then it proceed you to git installer page which looks like -

Simply click on download then git will start downloading and y'll be ready with git on your Mac OS.

Git version:

To check which version of git your are having in your system just run git --version command .

```
dikshamalik@Dikshas-Air ~ % git --version
git version 2.15.0
```

Now your OS is ready to run git commands.

Some basic Mac OS commands:

- *pwd command* : It prints the current working directory path.
- *cd command* : cd (Change directory) command is used to change the current working directory
- *ls command* : It is used to list information about files and directories within the file system.

Basic Git setup :

To set up your name use git command git config --global user.name “Name”

```
git config --global user.name "diksha6036"
```

To check whether the name is set or not just run command git config --global user.name

```
dikshamalik@Dikshas-Air Git.Github % git config --global user.name
diksha6036
```

To set up your email use git command git config --global user.email “email”

```
git config --global user.email "malikdiksha385@gmail.com"
```

To check whether the email is set or not just run command git config --global user.email

```
dikshamalik@Dikshas-Air Git.Github % git config --global user.email
malikdiksha385@gmail.com
```

Experiment No. 2

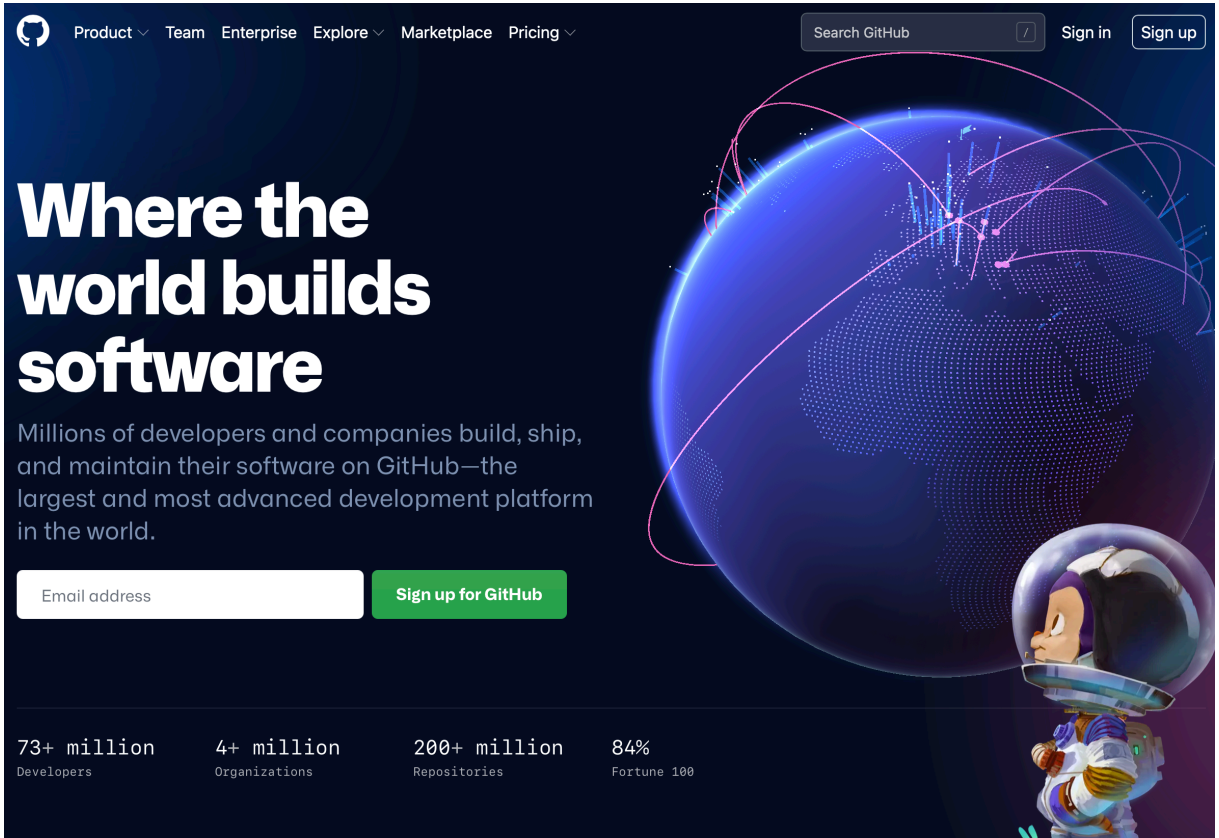
Aim: Setting up Git hub account.

What Is Github?

- Github is a website which hosts Git repositories.

Github Account:

To create an account on Github visit - https://github.com



Now follow the following steps to successfully create an account :

- Click on sign up
- Enter your mail and continue .

- Then create a strong password .

After doing all these steps y'll be having your GitHub account ready to work with.

Experiment No. 3

Aim: Program to generate logs.

Step 1:

Run git init command to create a new git repository .

```
dikshamalik@Dikshas-Air Git.Github % git init
Initialized empty Git repository in /Users/dikshamalik/Desktop/Git.Github/.git/
```

Using git init command again in the same directory will reinitialise your directory and can reset or undo the local changes to the state of git repository.

Step 2:

Run git status command to check are the files present in our working directory tracked by git or not. It lets you see which changes have been staged ,which haven't.

```
dikshamalik@Dikshas-Air Git.Github % git status
On branch master

No commits yet

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

        dikshamalik
        first.txt
        second.txt

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

Step 3:

Run git add <filename> command to add our directory in staging area which let's git know which files are going to be a part of next commit .

```
git add first.txt
```

To add all files in staging area use git command git add --a or git add .

Step 4:

Run git commit -m "any message here" to commit the files we've staged.

```
dikshamalik@Dikshas-Air Git.Github % git commit -m"Initial commit"
[master (root-commit) 513a8d2] Initial commit
 3 files changed, 22 insertions(+)
 create mode 100644 dikshamalik
 create mode 100644 first.txt
 create mode 100644 second.txt
```

Git commit command captures the changes which were made in the files that were staged. Now we can check whether any file is present in our staging area or not by running git status.

```
dikshamalik@Dikshas-Air Git.Github % git status
On branch master
nothing to commit, working tree clean
```

Step 5:

Run git log command to see all the commit of git repository and at which time the commit was Done and by whom it was made .

```
dikshamalik@Dikshas-Air Git.Github % git log
commit 513a8d26759f97d59a824920680855a1573df93c (HEAD -> master)
Author: diksha6036 <malikdiksha385@gmail.com>
Date:   Sun Apr 10 22:52:25 2022 +0530

    Initial commit
```

NOTE:

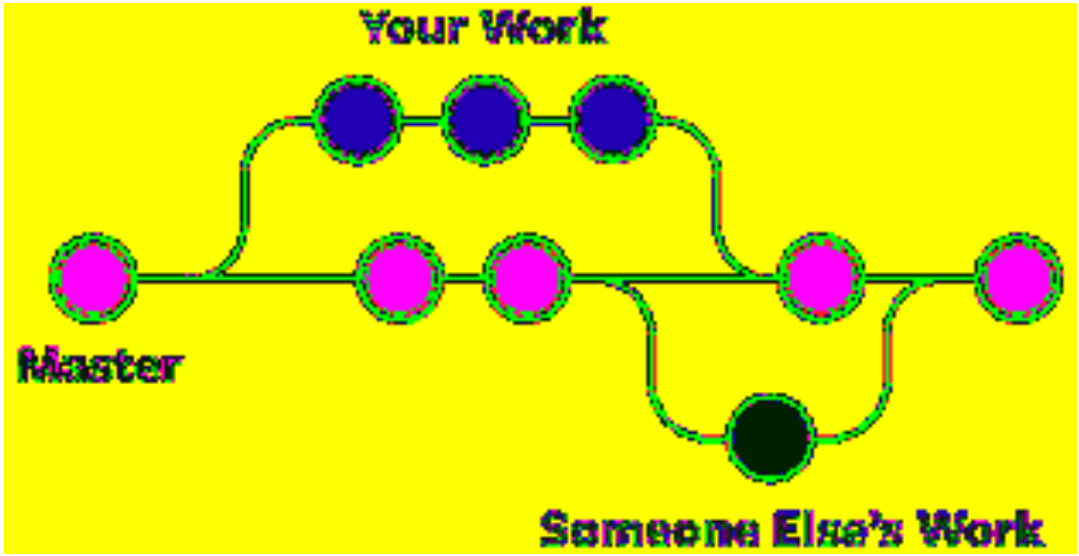
If you want to delete your whole git repository and git folder use rm -rf .git command.

Experiment No. 4

Aim : Create and Visualise branches in Git.

Git branch :

The main branch in git is called master branch.But we can make branches out of this main master branch.



Creating New branch:

- For creating a new branch use git branch “name of branch” Or git checkout -b “branch_name” commands

```
dikshamalik@Dikshas-Air Git.Github % git checkout -b "develop"
Switched to a new branch 'develop'
```

- To check how many branches we have use git branch command.

```
dikshamalik@Dikshas-Air Git.Github % git branch
* develop
  master
```

- To change the present working branch use git checkout “branch_name” command.

```
dikshamalik@Dikshas-Air Git.Github % git checkout master
Switched to branch 'master'
```

Visualizing Branches:

To visualise we have to create a new file in “develop” branch .
Add the file to staging area and commit.

```
dikshamalik@Dikshas-Air Git.Github % git checkout develop
Switched to branch 'develop'
dikshamalik@Dikshas-Air Git.Github % touch Myfile
dikshamalik@Dikshas-Air Git.Github % git status
On branch develop
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        Myfile

nothing added to commit but untracked files present (use "git add" to t
dikshamalik@Dikshas-Air Git.Github % git add .
dikshamalik@Dikshas-Air Git.Github % git commit -m"New branch commit"
[develop 86d1050] New branch commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Myfile
```

Now if you switch to previous i.e master branch y’ll not be able to find the file we made in develop branch. This is called the visualising of branch.

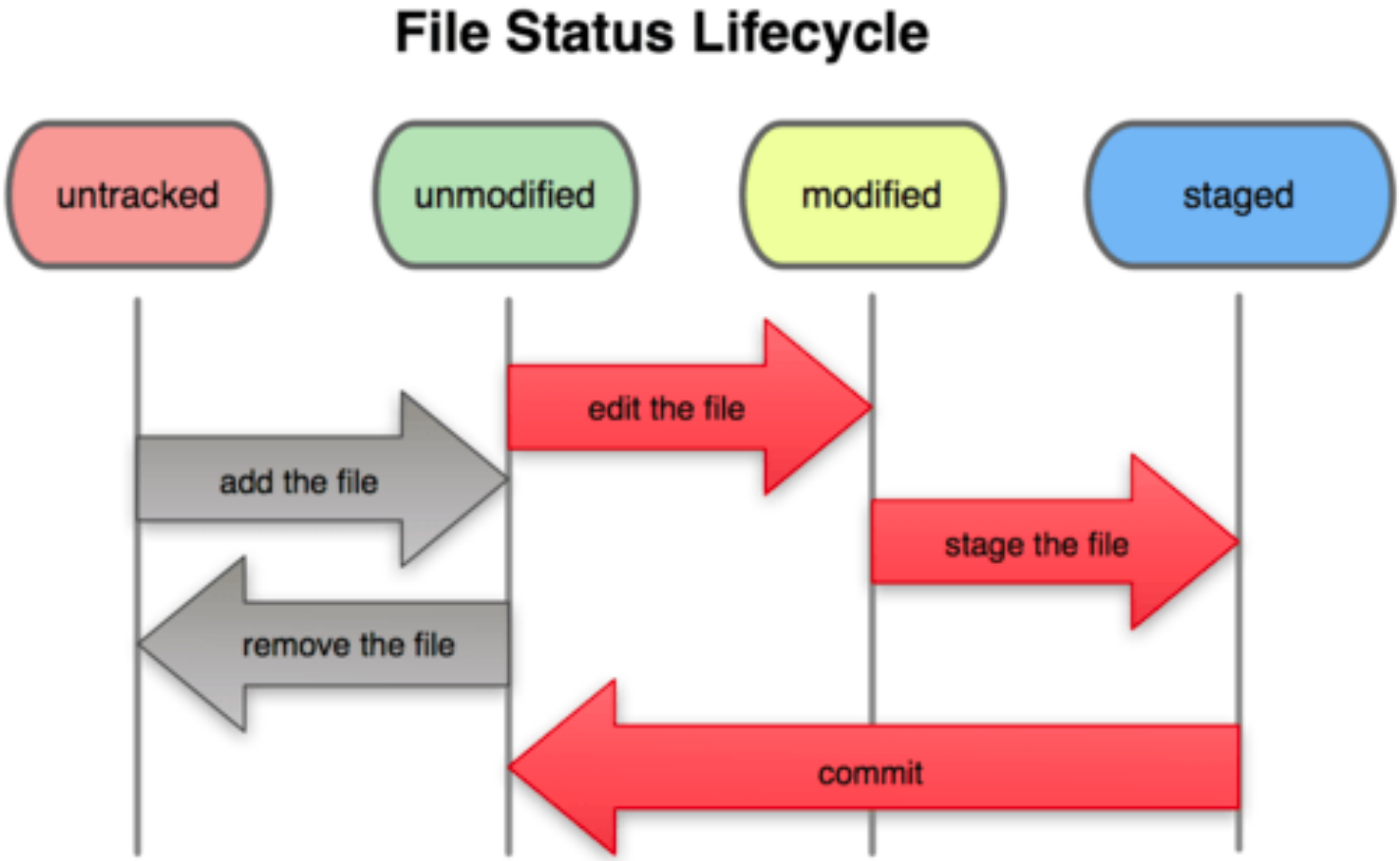
Lets try to find the file in master branch we've created in another branch :

```
dikshamalik@Dikshas-Air Git.Github % git checkout master
Switched to branch 'master'
dikshamalik@Dikshas-Air Git.Github % ls
dikshamalik     first.txt       second.txt
```

Note: We can also merge the two branches and can get the best results .

Experiment No. 5

Aim: Git life cycle description .



- If we put some files first in staging area and then modify a file and commit then git will ignore that modification and commit unmodified files.

- If we want our modified files to be commit then first we have to add that modified file in the staging area and then commit.

Git uses Three-tier Architecture:

→ Working directory

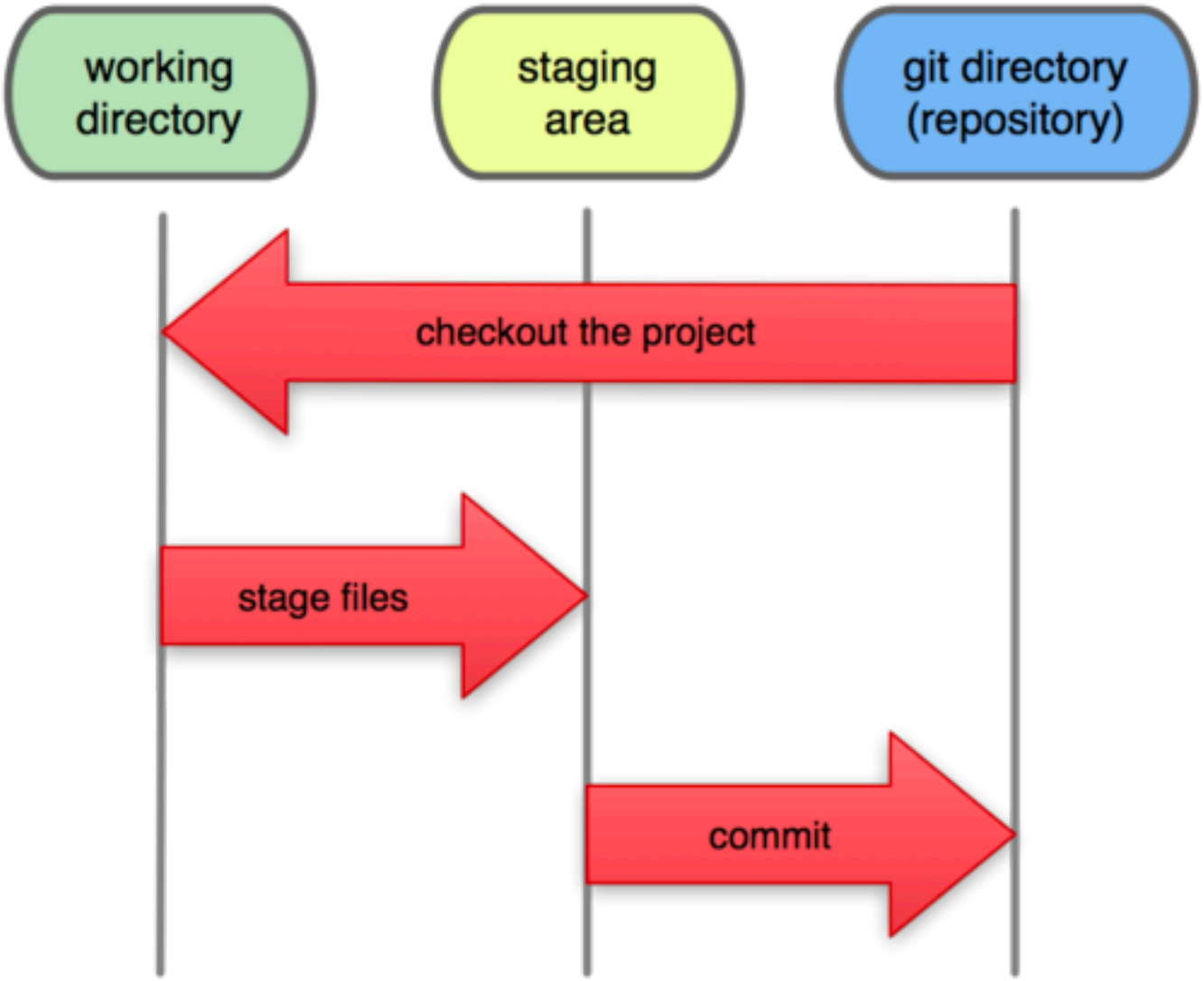
→ Staging area

→ Local repository

The three stages of git can store different or same stages of the same code in each Stage .

The below diagram represents these three states:

Local Operations



Working Directory:

Working directory specifies the folder where your files are stored .

Staging area :

Staging area is where your those files are present which you want to send to commit (to create snapshot of files)

After commit is fired , files which are in staging area will move to git repository

Git repository:

If you made any changes in files which are in git repository , those files (with Changes) will be in unstaging area . You again have to add them into staging area and commit.