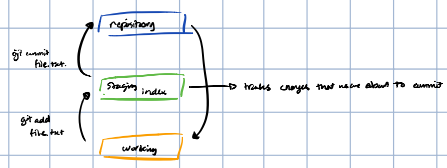
The three trees

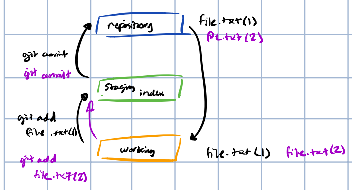
tree → file structure

- Typical two tree architecture

- THREE TREE ARCHITECTURE OF GIT:



git workflows



The HEAD pointer

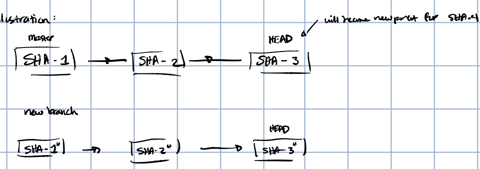
→ it points to / references a specific commit of the repository

points to the current branch of the repository

like a playback + record head

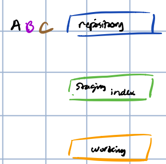
- as we record things the head moves along the cassette tape
- when we stop, we have the head posed on the last thing we have recorded
- if we rewind + record it will start recording over the previously recorded tape

Illustration:



Looks inside HEAD file (inside .git)

HASH values (SHA-1)



git generates **checksum** for each change set

- → changing data changes checksum
- → same data always equals checksum

checksum: a number fed into a mathematical algorithm to generate a simple #.

algorithm is SHA-1 hash algorithm in creating checksum

- 40 character hexadecimal string (0-9, a-f)
- contains the meta data
  - parent, author, message
    - ↓ commit before

Making changes to file

ADD FILES

- git status → will most likely tells:

"On branch master

nothing to commit

working tree is clean" → stuff in git repository matches working directory

"creates more file"

git status → new message "untracked files:"

git add new_files → now new_files is in staging area

git commit -m "new_files are now in project"

NOW new_file is in the repository

DELETE FILES

① git rm file_to_delete1.txt → also adds change to staging directory

② git commit -m "Delete first file"

Now the file no longer exists at the HEAD of the repository

EDIT FILES

- change text file in working directory so that it is different from what is in staging area + git repository
- have to stage the changed files + then commit so they are in repository

VIEW CHANGES W/ DIFF.

- make changes to file
- don't remember what they were
- DIFF to the rescue!
- git diff → sees changes btwn "a" staging tree + "b" working directory files
  - → shows what lines of code are different or new

VIEW ONLY STAGED CHANGES

- when you "git add" you can no longer track changes w/ the repository for a particular file
- to see differences btwn repository + staging:

"git diff --staged"

or

"git diff --cached"

MOVE AND RENAME FILES

- If you change the name:

METHOD 1

git status will say "the previously named file is gone!" AND "there is a new file!"

- will have to "git add new_named_file" AND "git rm old_named_file"

METHOD 2

"git mv old_named_file.txt new_named_file.txt" ← THE BETTER WAY!