Advance Git

** Interactive Rebase

↳ A tool for optimizing & cleaning up commit history

- change commit's message
- delete commits
- reorder commits
- Combine multiple commits into one
- edit/split an existing commit into multiple new ones.

* Procedure to apply Interactive Rebase

1. To determine the range of commits which is needed to be change (What should be base commit?)
2. git rebase -i Head~3
3. In editor, determine the actions to perform

reword → helps to change the commit message
squash → works by combining the line we mark up here with the one above.

Points to Remember:-

→ Commits manipulated will have new hash IDs (like new commit)

→ Do not use Interactive rebase on the commits that are already pushed/shared on remote repo.

→ Use it for cleaning up local commit history before merging.

→ Reflog contains information about the former state of branches and allows for reverting state if necessary

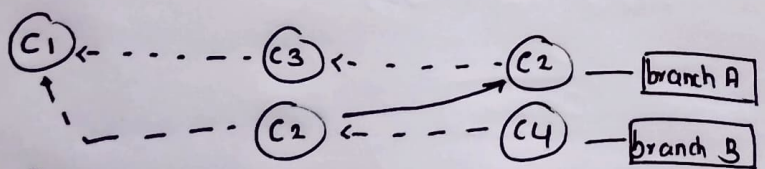
→ Reflog is used to:-
- Restore commits
- Restore deleted branches

→ Submodules are like repo under the parent repo

→ Submodules are we to keep the original work and third party content separate.

** Cherry picking:- (Integrating Single, Specific commit)

↳ cherry pick allows to select individual commits to be integrated.



(Here, we integrated C2 but not C4)

When to use?

→ when commit is accidently made to the wrong branch. You can swith to the correct branch & cherry pick the commit to where it should belong.

* Procedure to apply cherry pick :-

1. Copy the hash ID of the commit which is needed to cherry pick
2. Go to the branch, where it is needed to be integrated.
3. git cherry-pick <hash ID>
4. To delete the last commit from the branch :-
   Go to that branch using checkout
   git reset --hard HEAD~1

Reflog :-

↳ To keep track of changes made to branch tips

→ Reflog keep track of when git references in the local repository were modified.

→ A separate reflog is kept for git stash.

→ Reflogs are kept in folders under git directory of the local repository.

git reflog

Submodules :-

↳ fully functional $ Git repo. We can modify files commit, pull, push from inside.

git submodule add <git repo>

Search $ Find

(Filtering Commit History)

by date	--before / --after
by message	--grep
by author	--author
by file	-- <filename>
by branch	<branch A>

git log grep = "refactor"
git log --author = "hrishikesh 332"

git log --after = "2023-2-1"
--before = "2023-5-2"