git show: it is used to show the file names in log

command: git show commit_id --name-only

DEVOPS BY MUSTAFA

git show <commit> --stat : you'll see the commit summary along with the files that changed and details on how they changed.
git commit --amend -m "New commit message" : to edit the commit message
git commit --amend --no-edit : used to add some files in previous commit. (--no-edit means that the commit message does not change.)
git update-ref -d HEAD : used to delete all the commits in git
git reset commit: used to delete all the commits (upto the commit id)

git reset --hard HEAD~1 : used to delete latest commit along with the changes
git reset --hard HEAD~N : upto nth commit
git revert commit-id : used to undo a middle of the change (file also deleted)

Let's say that you forgot to configure the email and already did your first commit. Amend can be used to change the author of your previous commit as well. The author of the commit can be changed using the following command:

git commit --amend --author "Author Name <Author Email>"

GIT BRANCH:

- A branch represents an independent line of development.
- A branch is a way to isolate development work on a particular aspect of a project.
- The git branch command lets you create, list, rename, and delete branches.
- The default branch name in Git is master.

DEVOPS BY MUSTAFA

to see the current branch: git branch
to create a branch: git branch branch_name
to go to a branch: git checkout branch
to delete a branch: git branch -d branch_name
to rename a branch: git branch -m old new
to create and switch at a time: git checkout -b branch_name

GIT MERGE:

It allows us to get the code from one branch to another. This is useful when developers work on the same code and want to integrate their changes before pushing them up in a branch.

command: git merge branch_name

GIT MERGE CONFLICTS

Merge conflicts happen when you merge branches that have competing commits, and Git needs your help to decide which changes to incorporate in the final merge.

GIT makes merging super easy!
CONFLICTS generally araise when two people two people have changed the same lines in a file (or) if one developer deleted a file while another developer is working on the same file!
In this situation git cannot determine what is correct!

Lets understand in a simple way!

cat>file1 : hai all
add & commit
git checkout -b branch1
cat>file1 : 1234
add & commit
git checkout master
cat>>file1 : abcd
add & commit

git merge branch1 : remove it

DEVOPS BY MUSTAFA
DEVOPS BY MUSTAFA

identify merge conflicts:

see the file in master branch then you will see both the data in a single file including branch names

resolve:

open file in VIM EDITOR and delete all the conflict dividers and save it! add git to that file and commit it with the command (git commit -m "merged and resolved the conflict issue in abc.txt")

merge:

if you have 5 commits in master branch and only 1 commit in devops branch, to get all the commits from master branch to devops branch we can use merge in git. (command: git merge branch_name)

cherry-pick:

if you have 5 commits in master branch and only 1 commit in devops branch, to get specific commit from master branch to devops branch we can use cherry pick in git. (git cherry-pick commit_id).

DEVOPS BY MUSTAFA
DEVOPS BY MUSTAFA