|  |
| --- |
|  |
| 1) Git branch: |
|  | It is used when we have customized special needs for a particular task. We simply branch from the main course of project and create files that are needed to meet the purpose and saved. These changes are not necessasarily needed by the main project. |
|  | Syntax: |
|  | git branch (to verify how many branches exist ) |
|  | git branch branch\_name(to create a branch) |
|  | 2)git checkout: |
|  | this command is used to switch between different branches according to the requirement. |
|  | This command can also be used to use a version that was tagged previously. |
|  | Syntax: |
|  | git checkout branch\_name/version\_name |
|  | 3) merge: |
|  | This command is helpful when a developer wants to merge the changes made under the branch to the main project so the successful changes obtained in the branch can be reflected on the main project if the need be. |
|  | Syntax: |
|  | git merge branch\_name |
|  | 4) git ignore: |
|  | This is used when we want to block few files from entering the git repository. We create git ignore file using command: |
|  | touch .gitignore |
|  | Next we go to our local repository folder add the list of files that we want the git to ignore in that file. We add the git ignore file as follows: |
|  | git add .gitignore |
|  | Say I have mentioned a file name "file1.txt" inside my .gitignore file then when I create and add this file using git add it is simply ignored by the git. |
|  | 5)git push: |
|  | This command is used to push files from local repository to remote repository it is cloned with. This can be helpful as we can update a file in github from our local machine and push it back to the github. |
|  | syntax: |
|  | git push |
|  | 6)git stash: |
|  | This command is helpful when the programmer has made a few modifications in a file/code and has added it to staging area but is not willing to commit the changes at the moment. Then he can use : |
|  | git stash |
|  | and switch to another branch come back later to fetch the file and commit it using: |
|  | git stash apply |
|  | git commit |
|  | 7)git head: |
|  | This is used to get the recent information regarding the commitment made recently and also a head value that changes with every commit command given after modifying a file. |
|  | Syntax: |
|  | git show head |
|  | 8)git pull: |
|  | This command is used to get files from remote repository to our local repository. This is helpful when we want to fetch files in repository and make changes to the file accordingly. |
|  | Syntax |
|  | git pull <remote name created to clone explicitly> <branch-name> |
|  | 9)git remote: |
|  | This command is explicitly used to clone with a remote repository using an id like name. |
|  | Syntax: |
|  | git remote add <any\_name> <git repository url> |
|  | To check for the existing : |
|  | git remote -v |
|  | To remove: |
|  | git remote rm <name> |
|  | 10)git revert: |
|  | This command is helpful when we want to undo the previous commit made. For this we get access to the head value and type as: |
|  | git revert <head\_value> |
|  | 11)git tag: |
|  | This command is useful when we want to maintain different versions of the same file to meet different requirements. |
|  | Syntax: |
|  | git tag (This gives all the versions present in the repository) |
|  | git tag version\_number (creates a tag to the files that were added and committed previously) |
|  | git checkout version\_number (to fetch files stored under that tag) |
|  | 12) git mv: |
|  | This command is used when want to rename and move files in a repository. |
|  | Syntax: |
|  | git mv old\_file new\_file |
|  | 13)git init:\par |
|  | This is used for creating a new git repository. This is the first command that needs to be executed before other git commands. |
|  | Syntax: |
|  | git init |
|  | 14)git add: |
|  | This is used to move a file to a staged area before commiting the changes. |
|  | Syntax: |
|  | git add filename |
|  | 15)git commit: |
|  | This command is used to save changes to our local repository before pushing it to a remote repository. |
|  | 16) git clone: |
|  | This is used to link a remote repository from our local machine to push and pull files as and when we require. |
|  | Syntax: |
|  | git clone <Url of the repository> |
|  | 17) git restore: |
|  | This command is used to remove files from staged area after the git add command. |
|  | Syntax: |
|  | git restore --staged filename |
|  | 18) git diff: |
|  | This command can be used to check after modification of a file before using add command to spot the differences: |
|  | Syntax: |
|  | git diff |
|  | 19) git log: |
|  | This command is used to track and get information regarding the previous commits made and also information regarding who made the commits. |
|  | Syntax: |
|  | git log |
|  | 20) git status: |
|  | This tells us the status of the repository whether or not we have files to be added and commited. |
|  | Syntax: |
|  | git status |
|  | 21) |
|  | Username: |
|  | git config --global user.name "User Name" |
|  | User email: |
|  | git config --global user.email "Usermail@gmail.com" |
|  | 22)Local repository: This type of repository can be used when a person is working alone on a particular project. |
|  | Centralized repository: |
|  | Developers who work on a project can use it to collaborate. |
|  | Distributed Repository: |
|  | The user has a local copy of repository. This repository contains all the files and metadata present in the main repository. |
|  | 23) Features of git vcs |
|  | 1)Open source |
|  | 2)Scalable |
|  | 3)Distributed |
|  | 4)Security |
|  | 5)Speed |
|  | 24) Version control system is a software that keeps track of the changes that are made to a file in order to keep track. It enables developers to maintain different versions of the same software. |
|  | Benefits: |
|  | i) Saves time |
|  | ii) Performance |
|  | iii)Offline working |
|  | iv) Undo mistake |
|  | v) Track changes |
|  |  |
|  | 25) Git index is staging area between working directory and the repository . The modified files are stored using git add and git commit. |
|  |  |
|  |  |