GitHub commands and their usage

1. **git init** : Initializes a new Git repository.
2. **git clone [url]** : Clones a repository from a remote URL.
3. **git status** : Displays the state of the working directory and the staging area.
4. **git add [file]** : Adds a file to the staging area.
5. **git add .** : Adds all files to the staging area.
6. **git commit -m "[message]"** : Commits the staged changes with a message.
7. **git push** : Pushes the local changes to the remote repository.
8. **git pull** : Fetches and merges changes from the remote repository.
9. **git branch** : Lists all the branches in the repository.
10. **git branch [branch-name]** : Creates a new branch.
11. **git checkout [branch-name]** : Switches to the specified branch.
12. **git merge [branch-name]** : Merges the specified branch into the current branch.
13. **git log** : Shows the commit history.
14. **git remote add origin [url]** : Adds a remote repository.
15. **git remote -v** : Displays the remote URLs.
16. **git fetch** : Downloads objects and refs from another repository.
17. **git reset [file]** : Unstages a file.
18. **git reset --hard** : Resets the index and working directory to the last commit.
19. **git rm [file]** : Removes a file from the working directory and the staging area.
20. **git stash** : Temporarily saves changes that are not ready to be committed.
21. **git stash apply** : Applies the stashed changes.

22. **git tag [tag-name]** : Creates a new tag.
23. **git diff** : Shows changes between commits, commit and working tree, etc.
24. **git rebase [branch-name]** : Reapplies commits on top of another base tip.
25. **git cherry-pick [commit-id]** : Applies the changes introduced by an existing commit.
26. **git mv [old-filename] [new-filename]** : Renames or moves a file.
27. **git blame [file]** : Shows what revision and author last modified each line of a file.
28. **git show [commit-id]** : Displays information about a specific commit.
29. **git log --oneline** : Shows a concise commit history.
30. **git shortlog** : Summarizes git log output.
31. **git reflog** : Shows a log of changes to the local repository's reference history.
32. **git ls-files** : Lists all files in the index.
33. **git clean -f** : Removes untracked files from the working directory.
34. **git bisect start** : Starts a bisect session to find the commit that introduced a bug.
35. **git bisect good [commit-id]** : Marks a commit as good in the bisect process.
36. **git bisect bad [commit-id]** : Marks a commit as bad in the bisect process.
37. **git archive --format=zip --output=[filename.zip] [branch-name]** : Creates an archive of files from a named tree.
38. **git rev-parse [branch-name]** : Parses and displays various information about revisions.
39. **git cherry [branch1] [branch2]** : Shows commits in branch1 that are not in branch2.
40. **git tag -d [tag-name]** : Deletes a tag from the repository.

41. **git config --global user.name "[name]"** : Sets the name to be used in all projects on the system.
42. **git config --global user.email "[email]"** : Sets the email to be used in all projects on the system.
43. **git submodule add [url] [path]** : Adds a new submodule to the repository.
44. **git submodule update --init** : Clones and initializes all the submodules.
45. **git remote remove [name]** : Removes a remote repository.