**1. Configuring Git**

* git config --global user.name "Your Name": Sets the username for Git.
* git config --global user.email "youremail@example.com": Sets the email for Git.

### 2. ****Initializing a Repository****

* git init: Initializes a new Git repository in the current directory.

### 3. ****Cloning a Repository****

* git clone <repository-url>: Clones an existing repository from a remote source to your local machine.

### 4. ****Checking the Status****

* git status: Shows the current status of the working directory and staging area (e.g., modified files, new files).

### 5. ****Adding Changes****

* git add <file>: Adds a specific file to the staging area.
* git add .: Adds all modified and new files to the staging area.

### 6. ****Committing Changes****

* git commit -m "Commit message": Commits the staged changes with a descriptive message.

### 7. ****Viewing the Commit History****

* git log: Shows the commit history in the current branch.
* git log --oneline: Shows the commit history in a more compact, one-line format.

### 8. ****Branching and Merging****

* git branch: Lists all branches in the repository.
* git branch <branch-name>: Creates a new branch.
* git checkout <branch-name>: Switches to the specified branch.
* git merge <branch-name>: Merges the specified branch into the current branch.

### 9. ****Pushing and Pulling****

* git push: Pushes local commits to the remote repository.
* git push origin <branch-name>: Pushes a specific branch to the remote repository.
* git pull: Fetches and merges changes from the remote repository to your local branch.

### 10. ****Stashing Changes****

* git stash: Stashes the current changes (saving them temporarily and restoring the working directory to the last commit).
* git stash pop: Restores the most recent stash and removes it from the stash list.
* git stash list: Lists all stashes.

### 11. ****Resetting Changes****

* git reset: Unstages files from the staging area but leaves the working directory unchanged.
* git reset --hard: Resets both the staging area and working directory to the last commit.

### 12. ****Remote Repositories****

* git remote -v: Lists the remote repositories associated with the current project.
* git remote add <name> <url>: Adds a new remote repository.
* git fetch: Fetches changes from the remote repository without merging.

### 13. ****Tagging****

* git tag <tag-name>: Creates a new tag at the current commit.
* git tag: Lists all tags in the repository.

### 14. ****Undoing Changes****

* git checkout -- <file>: Discards changes in the working directory (for a specific file).
* git revert <commit-id>: Creates a new commit that undoes the changes made in the specified commit.

### 15. ****Viewing Differences****

* git diff: Shows the differences between the working directory and the staging area.
* git diff <commit-id> <commit-id>: Shows the difference between two commits.

Let me know if you'd like more details or a deeper explanation of any specific commands!

##### You said:

git all basic commands

###### ChatGPT said:

Certainly! Here's a comprehensive list of basic Git commands, categorized for clarity:

**1. Configuration Commands**

* git config --global user.name "Your Name": Sets your username for Git commits.
* git config --global user.email "youremail@example.com": Sets your email address for Git commits.
* git config --global core.editor "editor": Sets your default text editor for Git.

**2. Repository Setup**

* git init: Initializes a new Git repository in the current directory.
* git clone <repository-url>: Creates a copy of an existing repository from a remote source.

**3. Working with Files**

* git add <file>: Stages a specific file for commit.
* git add .: Stages all modified and new files in the current directory.
* git commit -m "Commit message": Commits the staged changes with a descriptive message.
* git status: Displays the current state of the working directory and staging area.
* git diff: Shows the differences between the working directory and the staging area.
* git diff <commit-id> <commit-id>: Shows the differences between two commits.

**4. Branching and Merging**

* git branch: Lists all branches in the repository.
* git branch <branch-name>: Creates a new branch.
* git checkout <branch-name>: Switches to the specified branch.
* git checkout -b <branch-name>: Creates and switches to a new branch.
* git merge <branch-name>: Merges the specified branch into the current branch.
* git branch -d <branch-name>: Deletes the specified branch.

**5. Remote Repositories**

* git remote -v: Lists the remote repositories associated with the current project.
* git remote add <name> <url>: Adds a new remote repository.
* git fetch: Fetches changes from the remote repository without merging.
* git pull: Fetches and merges changes from the remote repository to your local branch.
* git push: Pushes local commits to the remote repository.
* git push origin <branch-name>: Pushes a specific branch to the remote repository.

**6. Viewing Commit History**

* git log: Shows the commit history in the current branch.
* git log --oneline: Shows the commit history in a more compact, one-line format.
* git show <commit-id>: Displays detailed information about a specific commit.

**7. Undoing Changes**

* git reset: Unstages files from the staging area but leaves the working directory unchanged.
* git reset --hard: Resets both the staging area and working directory to the last commit.
* git revert <commit-id>: Creates a new commit that undoes the changes made in the specified commit.
* git checkout -- <file>: Discards changes in the working directory for a specific file.

**8. Stashing Changes**

* git stash: Stashes the current changes, saving them temporarily and restoring the working directory to the last commit.
* git stash pop: Restores the most recent stash and removes it from the stash list.
* git stash list: Lists all stashes.

**9. Tagging**

* git tag <tag-name>: Creates a new tag at the current commit.
* git tag: Lists all tags in the repository.

**10. Cleaning Untracked Files**

* git clean -f: Removes untracked files from the working directory.
* git clean -fd: Removes untracked files and directories.

**11. Viewing Differences**

* git diff: Shows the differences between the working directory and the staging area.
* git diff <commit-id> <commit-id>: Shows the differences between two commits.