Workflow of Git Commands

- **`git --version`**: Check the Git version.

- **`git init`**: Initialize an empty folder to track files.

- **Configure User Details:**

    - `git config user.name "Your UserName"`: Configure the username locally.
    - `git config user.email "Your Email"`: Configure the user's email locally.
    - `git config --global user.name "Your UserName"`: Configure the username globally.
    - `git config --global user.email "Your Email"`: Configure the user's email globally.
- **Staging Files:**

    - `git add filename`: Add a particular file to the staging area.
    - `git add --all` / `git add .`: Add all files to the staging area.
    - `git add *.extension`: Add files of a specific extension (e.g., `.java`, `.py`).
- **Committing Files:**

    - `git commit -m "Commit Message"`: Commit files from staging to the local repository.
    - `git commit --amend`: Modify the last commit (use with caution).
- **Remote Repository:**

    - `git remote add origin GitHubURL`: Establish a connection between the local and remote repositories.
    - `git remote -v`: Verify the connection to the remote repository.
    - `git push origin branch_name`: Upload changes from the local repository to the remote (default branch: `master`).
- **Status and Logs:**

    - `git status`: Check the current status of the working directory and staging area.
    - `git log`: View commit history.
    - `git log --oneline`: View concise commit history.
    - `git diff`: Show changes between the working directory and staging area.
    - `git diff commit_id1 commit_id2`: Compare two specific commits.

Branching Commands

- **Creating and Managing Branches:**
    - `git branch branch_name`: Create a new branch locally.
    - `git branch`: List all local branches.
    - `git branch --list`: List all local branches.
    - `git branch -r`: List remote branches.
    - `git branch -a`: List both local and remote branches.
- **Switching and Creating Branches:**
    - `git checkout branch_name` / `git switch branch_name`: Switch to a branch.
    - `git checkout -b branch_name`: Create and switch to a branch simultaneously.
- **Renaming and Deleting Branches:**
    - `git branch -m oldbranchname newbranchname`: Rename a branch.
    - `git branch -d branch_name`: Delete a local branch.
    - `git branch -D branch_name`: Force delete a local branch.
    - `git push origin --delete branch_name`: Delete a remote branch.
- **Merging and Cherry-Picking:**
    - `git merge branch_name`: Merge a branch into the current branch.
    - `git cherry-pick commit_id`: Apply a specific commit to another branch.
- **Uploading Branches:**
    - `git push origin branch_name`: Push changes to a branch on the remote repository.

Undoing Changes

- **Undoing in the Working Directory:**
    - `git checkout filename`: Undo changes to a specific file in the working directory.
    - `git checkout .`: Undo changes to all files in the working directory.

- **Removing Files:**
    - `git rm filename`: Remove a file from the local repository.
    - `git rm --cached filename`: Remove a file from the staging area (keep it locally).
- **Editing Configuration:**
    - `git config --edit`: Edit configuration details like email and username.
- **Reverting Commits:**
    - `git revert commit_id`: Create a new commit that undoes changes made in a specific commit.

Other Useful Commands

- `git clone GitHubURL`: Clone a remote repository to your local system.
- `git pull origin branch_name`: Fetch and merge changes from the remote repository to the local branch.
- `git stash`: Save changes temporarily and clean the working directory.
- `git stash pop`: Reapply the stashed changes.
- `git reflog`: View reference logs, including commits that were reset or deleted.
- `git reset HEAD filename`: Unstage a specific file.
- `git reset --soft commit_id`: Reset to a commit while keeping changes staged.
- `git reset --hard commit_id`: Reset to a commit and discard all changes.
