Git Cheat Sheet

1. Git Setup & Configuration:

```
git config --global user.name "Your Name" # Set global username
git config --global user.email "you@example.com" # Set global email
git config --global core.editor "vim" # Set default Git editor
git config --list # View global Git settings
git config --global alias.st status # Create Git alias
```

2. Initializing & Cloning Repositories:

```
git init # Initialize a new Git repository
git clone <repo-url> # Clone a remote repository
git clone --depth=1 <repo-url> # Clone with minimal history
```

3. Staging & Committing Changes:

```
git status # Show current changes
git add <file> # Stage a specific file
git add . # Stage all changes
git commit -m "message" # Commit staged changes
git commit --amend -m "New message" # Modify last commit message
git commit -am "msg" # Add & commit all tracked files in one step
```

4. Branching & Merging:

```
git status # Show current changes
git add <file> # Stage a specific file
git add . # Stage all changes
git commit -m "message" # Commit staged changes
git commit --amend -m "New message" # Modify last commit message
git checkout -b <branch> # Create and switch to a new branch
git checkout - # Switch back to the last branch
git branch branch-name # create new branch
git branch -m old-branch-name new-branch-name # Rename branch name
```

5. Undoing & Fixing Mistakes:

```
git reset --soft HEAD~1 # Undo last commit (keep changes staged)
git reset --mixed HEAD~1 # Undo last commit (keep changes unstaged)
git reset --hard HEAD~1 # Undo last commit completely (delete changes)
git checkout -- <file> # Discard local changes to a file
git revert <commit-hash> # Create a new commit to undo a previous commit
git reset --hard origin/<branch> # Reset local branch to match remote
```

6. Working with Remote Repositories:

```
git remote -v # Show remote repositories
git remote add origin <url> # Add a remote repository
git remote set-url origin <new-url> # Change remote URL
git push origin <branch> # Push branch to remote
git push --force-with-lease # Force push safely
git fetch --all # Fetch latest changes from all remotes
git pull origin <branch> # Pull latest changes
git pull --rebase origin <branch> # Pull with rebase
```

7. Stashing Changes:

```
git stash # Save changes without committing
git stash list # Show list of stashed changes
git stash pop # Apply last stashed changes
git stash drop # Remove last stash
git stash clear # Remove all stashes
git clean -df # Remove untracked files and directories
```

8. Working with Tags:

```
git tag # List all tags
git tag <tag-name> # Create a lightweight tag
git tag -a <tag-name> -m "message" # Create an annotated tag
git push origin <tag-name> # Push a specific tag
git push --tags # Push all tags
git tag -d <tag-name> # Delete local tag
git push origin --delete <tag-name> # Delete remote tag
```

9. Viewing History & Logs:

```
git log # Show commit history
git log --oneline # Show commit history in one line per commit
git log --graph --decorate --all --oneline # Compact and visual commit history
git log --author="Your Name" # Show commits by author
```

```
git log --since="2 weeks ago" # Show commits from last 2 weeks
git show <commit-hash> # Show details of a specific commit
git blame <file> # Show who changed each line of a file
```

10. Cherry-Picking (Applying Specific Commits):

```
git cherry-pick <commit-hash> # Apply a specific commit from another branch
git cherry-pick --no-commit <commit-hash> # Apply changes without committing
git cherry-pick -x <commit-hash> # Apply commit with reference to original
```

11. Interactive Rebasing & Squashing Commits:

```
git rebase <branch> # Reapply commits on top of another branch
git rebase -i HEAD~3 # Interactive rebase for last 3 commits
git rebase --abort # Cancel an ongoing rebase
git rebase --skip # Skip a conflicting commit during rebase
```

12. Sync Fork with Original Repository:

```
git remote add upstream <original-repo-url>
git fetch upstream
git merge upstream/main
git push origin main
```