Git Ninja Handbook

90+ commands fully listed and categorized
Covers basic → intermediate → advanced → collaboration → submodules → worktrees → patches

1. Setup & Configuration

```
$ git --version                      # Show installed Git version
$ git help                           # Show general Git help
$ git config --list                   # Display all Git configs
$ 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 editor
$ git config --global alias.st status         # Alias "st" for status
$ git config --global alias.co checkout       # Alias "co" for checkout
$ git config --global alias.lg "log --oneline --graph --decorate --all" #
   Pretty log
```

2. Getting Started

```
$ git init               # Initialize a new repository
$ git clone <url>        # Clone remote repo
$ git remote -v          # Show remotes
$ git remote add origin <url> # Add new remote
$ git remote remove origin    # Remove a remote
```

3. Staging & Committing

```
$ git status             # Show repo status
$ git add <file>         # Stage a file
$ git add .              # Stage all files
$ git reset <file>       # Unstage a file
$ git commit -m "msg"    # Commit staged changes
$ git commit -am "msg"   # Stage + commit tracked files
$ git commit --amend     # Edit last commit
$ git log                # Show commit history
$ git log --oneline      # Short log
$ git log --graph --oneline --decorate --all  # Graph view
```

4. Branching & Merging

```
$ git branch               # List branches
$ git branch <name>        # Create new branch
$ git checkout <branch>    # Switch to branch
$ git checkout -b <branch># Create & switch branch
$ git switch <branch>      # Modern switch
$ git merge <branch>       # Merge branch into current
$ git rebase <branch>      # Reapply commits on top
$ git branch -d <branch>   # Delete branch
$ git branch -m newname    # Rename branch
$ git cherry-pick <commit># Apply specific commit
```

5. Working with Remotes

```
$ git fetch                  # Download new data
$ git fetch --prune          # Clean deleted branches
$ git pull                   # Fetch + merge
$ git pull --rebase          # Fetch + rebase
$ git push                   # Push changes
$ git push origin <branch>   # Push specific branch
$ git push -u origin <branch> # Push + set upstream
$ git push --force           # Force push (    overwrites)
$ git push origin --delete <branch> # Delete remote branch
```

6. Undoing Changes

```
$ git restore <file>           # Discard local changes
$ git restore --staged <file> # Unstage file
$ git reset --hard HEAD        # Reset working dir
$ git reset --soft HEAD~1      # Undo last commit keep staged
$ git revert <commit>          # Create new commit to undo
$ git checkout <commit>        # Checkout commit (detached)
$ git reflog                   # History of HEAD changes
$ git clean -fd                # Remove untracked files/dirs
```

7. Stashing

```
$ git stash                  # Stash changes
$ git stash list             # Show stashes
$ git stash show             # Show last stash
$ git stash apply            # Apply last stash
$ git stash pop              # Apply + drop last stash
$ git stash drop             # Drop last stash
$ git stash clear            # Clear all stashes
```

8. Tags & Releases

```
$ git tag                    # List tags
$ git tag <tag>              # Create lightweight tag
$ git tag -a <tag> -m "msg" # Create annotated tag
$ git show <tag>             # Show tag details
$ git push origin <tag>      # Push tag
$ git push origin --tags     # Push all tags
$ git tag -d <tag>           # Delete local tag
$ git push origin :refs/tags/<tag> # Delete remote tag
```

9. Collaboration

```
$ git pull origin main       # Pull main branch
$ git push origin main       # Push main branch
$ git push --set-upstream origin <branch> # Track branch
$ git cherry-pick <commit>   # Apply specific commit
$ git diff <branch1>..<branch2> # Show diff between branches
$ git merge --abort          # Abort merge in progress
$ git rebase --abort         # Abort rebase in progress
$ git rebase -i HEAD~3       # Interactive rebase last 3 commits
```

```
$ git notes add -m "msg"     # Add note to commit
$ git notes show <commit>    # Show notes
```

10. Advanced Git

```
$ git diff                   # Show unstaged changes
$ git diff --staged          # Show staged changes
$ git blame <file>           # Show who changed lines
$ git bisect start           # Start bug search
$ git bisect bad             # Mark commit bad
$ git bisect good <commit>   # Mark commit good
$ git submodule add <url>    # Add submodule
$ git submodule update --init --recursive # Init submodules
$ git worktree add ../dir branch # Work with multiple dirs
$ git gc                     # Garbage collection
$ git fsck                   # Integrity check
$ git shortlog -sn           # Show commit counts per author
$ git archive --format=zip HEAD > latest.zip # Export repo
$ git bundle create repo.bundle --all # Create bundle
$ git bundle verify repo.bundle        # Verify bundle
$ git bundle list-heads repo.bundle    # List heads in bundle
$ git apply patch.diff         # Apply patch file
$ git am <patch>               # Apply patch as commit
$ git filter-branch            # Rewrite history (deprecated)
$ git reflog expire --expire=now --all # Clean reflog
$ git credential-cache          # Cache credentials temporarily
$ git credential-store          # Store credentials permanently
$ git mergetool                # Launch merge tool
$ git difftool                 # Launch diff tool
$ git log --stat               # Show log with file stats
$ git log --pretty=oneline     # Compact commit log
$ git log --graph --all        # Graph all commits
$ git describe                 # Describe current commit
$ git reset --merge HEAD       # Reset while preserving working tree
```