

# Git Cheatsheet

By Dejan Panovski • Updated on Aug 2, 2026 •

## Quick reference for common Git commands and workflows

Git is a distributed version control system used to track changes in source code. This cheatsheet covers everyday commands for working with repositories, branches, commits, and remote collaboration.

### Getting Started

Initialize and configure Git.

|                                                     |                             |
|-----------------------------------------------------|-----------------------------|
| <code>git init</code>                               | Initialize a new repository |
| <code>git clone url</code>                          | Clone a remote repository   |
| <code>git status</code>                             | Show working tree status    |
| <code>git config --global user.name "Name"</code>   | Set global name             |
| <code>git config --global user.email "email"</code> | Set global email            |

### Staging and Committing

Add changes and create commits.

|                                      |                       |
|--------------------------------------|-----------------------|
| <code>git add file</code>            | Stage a file          |
| <code>git add .</code>               | Stage all changes     |
| <code>git commit -m "message"</code> | Commit staged changes |
| <code>git commit --amend</code>      | Amend last commit     |
| <code>git revert commit</code>       | Revert a commit       |

## Branches

Create, list, and switch branches.

|                                     |                       |
|-------------------------------------|-----------------------|
| <code>git branch</code>             | List branches         |
| <code>git branch name</code>        | Create a branch       |
| <code>git switch name</code>        | Switch branch         |
| <code>git switch -c name</code>     | Create and switch     |
| <code>git branch -d name</code>     | Delete local branch   |
| <code>git branch -D name</code>     | Force delete branch   |
| <code>git branch -m new-name</code> | Rename current branch |

## Merge and Rebase

Combine branches and rewrite history.

|                                       |                           |
|---------------------------------------|---------------------------|
| <code>git merge branch</code>         | Merge branch into current |
| <code>git merge --no-ff branch</code> | Merge with merge commit   |
| <code>git merge --abort</code>        | Abort merge in progress   |
| <code>git rebase branch</code>        | Rebase onto branch        |
| <code>git rebase -i HEAD~3</code>     | Interactive rebase last 3 |
| <code>git rebase --abort</code>       | Abort rebase in progress  |
| <code>git cherry-pick commit</code>   | Apply a single commit     |

## Remote Repositories

Work with remotes and synchronization.

|                                              |                       |
|----------------------------------------------|-----------------------|
| <code>git remote -v</code>                   | List remotes          |
| <code>git remote add origin url</code>       | Add remote            |
| <code>git remote set-url origin url</code>   | Change remote URL     |
| <code>git remote remove origin</code>        | Remove remote         |
| <code>git fetch</code>                       | Fetch from remote     |
| <code>git pull</code>                        | Fetch and merge       |
| <code>git push</code>                        | Push to remote        |
| <code>git push -u origin branch</code>       | Push and set upstream |
| <code>git push origin --delete branch</code> | Delete remote branch  |

## Stash

Temporarily save uncommitted changes.

|                                          |                         |
|------------------------------------------|-------------------------|
| <code>git stash</code>                   | Stash changes           |
| <code>git stash push -m "message"</code> | Stash with message      |
| <code>git stash list</code>              | List stashes            |
| <code>git stash pop</code>               | Apply and remove latest |
| <code>git stash apply stash@{0}</code>   | Apply without removing  |
| <code>git stash drop stash@{0}</code>    | Delete a stash          |
| <code>git stash clear</code>             | Delete all stashes      |

## Worktrees

Check out multiple branches at once.

|                                                              |                                         |
|--------------------------------------------------------------|-----------------------------------------|
| <code><a href="#">git worktree add ../path branch</a></code> | Check out a branch in a linked worktree |
| <code>git worktree add -b name ../path main</code>           | Create a branch in a new worktree       |
| <code>git worktree add --detach ../path main</code>          | Detached worktree for testing           |
| <code>git worktree list</code>                               | List main and linked worktrees          |
| <code>git worktree lock ../path</code>                       | Protect a worktree from pruning         |
| <code>git worktree remove ../path</code>                     | Delete and unregister a worktree        |
| <code>git worktree prune</code>                              | Remove stale worktree entries           |

## History and Diff

Inspect history and changes.

|                                        |                               |
|----------------------------------------|-------------------------------|
| <code>git log</code>                   | View commit history           |
| <code>git log --oneline --graph</code> | Compact graph view            |
| <code>git log -p file</code>           | History of a file             |
| <code>git show commit</code>           | Show a commit                 |
| <code>git blame file</code>            | Show who changed each line    |
| <code>git reflog</code>                | Show reference log (recovery) |
| <code>git diff</code>                  | Unstaged diff                 |
| <code>git diff --staged</code>         | Staged diff                   |
| <code>git diff branch1..branch2</code> | Compare branches              |

## Undo and Cleanup

Discard or reset changes safely.

|                                        |                             |
|----------------------------------------|-----------------------------|
| <code>git restore file</code>          | Discard local changes       |
| <code>git restore --staged file</code> | Unstage a file              |
| <code>git reset --soft HEAD~1</code>   | Undo commit, keep changes   |
| <code>git reset --hard HEAD</code>     | Reset to last commit        |
| <code>git clean -fd</code>             | Remove untracked files/dirs |

## Tags

Create and manage tags.

|                                              |                   |
|----------------------------------------------|-------------------|
| <code>git tag</code>                         | List tags         |
| <code>git tag v1.0.0</code>                  | Create tag        |
| <code>git tag -a v1.0.0 -m "msg"</code>      | Annotated tag     |
| <code>git push --tags</code>                 | Push all tags     |
| <code>git tag -d v1.0.0</code>               | Delete local tag  |
| <code>git push origin --delete v1.0.0</code> | Delete remote tag |

## .gitignore

Exclude files from version control.

|                                   |                         |
|-----------------------------------|-------------------------|
| <code>*.log</code>                | Ignore all .log files   |
| <code>node_modules/</code>        | Ignore directory        |
| <code>!important.log</code>       | Negate ignore rule      |
| <code>**/build</code>             | Ignore in any directory |
| <code>git rm --cached file</code> | Untrack a tracked file  |