Git Commands Cheatsheet

The most-used Git commands with plain-English explanations and how engineers describe them in conversation.

https://coderslingo.com/resources/cheatsheets/git-commands-reference/

Basics

The everyday commands you run dozens of times a day.

`git clone <url>`
Copy a remote repository to your machine.
How engineers say it: "Go ahead and clone the repo."

`git status`
Show which files are changed, staged, or untracked.
How engineers say it: "What does git status say?"

`git add <file>`
Stage changes so they'll be in the next commit.
How engineers say it: "Stage the files you want to commit."

`git add .`
Stage all changes in the current directory.
How engineers say it: "Just add everything."

`git commit -m "message"`
Save staged changes as a snapshot with a message.
How engineers say it: "Commit it with a clear message."

`git push`
Upload your local commits to the remote.
How engineers say it: "Push your branch and open a PR."

`git pull`
Fetch and merge the latest changes from the remote.
How engineers say it: "Pull main before you start."

`git log --oneline`
Show commit history, one line each.
How engineers say it: "Check the log to see what landed."

Branching & Merging

Working on parallel lines of development and bringing them back together.

`git branch`
List branches (or create one with a name).
How engineers say it: "What branch are you on?"

`git switch <branch>`
Move to an existing branch (modern alternative to checkout).
How engineers say it: "Switch to the feature branch."

`git switch -c <branch>`
Create a new branch and switch to it.
How engineers say it: "Cut a new branch off main."

`git checkout <branch>`
Switch branches (older, broader command).
How engineers say it: "Check out the release branch."

`git merge <branch>`
Combine another branch's history into the current one.
How engineers say it: "Merge it into main."

`git rebase <branch>`
Replay your commits on top of another branch for a linear history.
How engineers say it: "Rebase onto main to clean up the history."

`git rebase -i HEAD~3`
Interactively edit, reorder, or squash recent commits.

How engineers say it: "Squash these commits into one."

Undoing & Stashing

Backing out of changes — safely or destructively.

`git restore <file>`
Discard uncommitted changes to a file.
How engineers say it: "Just restore the file to drop those changes."

`git reset <file>`
Unstage a file (keep the changes).
How engineers say it: "Un-stage that — reset it."

`git reset --hard <commit>`
Move the branch and discard all changes after a commit. Destructive.
How engineers say it: "Hard-reset to the last good commit." (careful!)

`git revert <commit>`
Create a new commit that undoes a previous one. Safe for shared history.
How engineers say it: "Revert the bad commit instead of force-pushing."

`git stash`
Shelve uncommitted changes to get a clean working tree.
How engineers say it: "Stash your changes and switch branches."

`git stash pop`
Re-apply the most recently stashed changes.
How engineers say it: "Pop the stash when you're back."

Inspecting

Looking at what changed, and who changed it.

`git diff`
Show unstaged changes line by line.
How engineers say it: "Let's look at the diff."

`git diff --staged`
Show changes that are staged for the next commit.
How engineers say it: "Check the staged diff before committing."

`git blame <file>`
Show who last changed each line and in which commit.
How engineers say it: "Git blame says this was changed last month."

`git show <commit>`
Show the full details and diff of a single commit.
How engineers say it: "Show me that commit."

Collaboration

Syncing with remotes and integrating work across the team.

`git fetch`
Download remote changes without merging them.
How engineers say it: "Fetch first so you can see what's new."

`git remote -v`
List the remotes and their URLs.
How engineers say it: "What does git remote point at?"

`git cherry-pick <commit>`
Apply a single commit from one branch onto another.
How engineers say it: "Cherry-pick that fix onto the release branch."

`git tag v1.2.0`
Mark a specific commit, usually for a release.
How engineers say it: "Tag the release and push the tags."