git add

Moves changes from the working directory to the staging area. This gives you the opportunity to prepare a snapshot before committing it to the official history.

**Related Tutorials**

[Saving changes: git add](https://www.atlassian.com/git/tutorials/saving-changes?section=git-add)[Learn Git with Bitbucket Cloud: Copy your Git repository and add files](https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud?section=copy-and-add-files)[Using Branches: git merge](https://www.atlassian.com/git/tutorials/git-merge)[Inspecting a repository: git status](https://www.atlassian.com/git/tutorials/inspecting-a-repository?section=git-status)

git branch

This command is your general-purpose branch administration tool. It lets you create isolated development environments within a single repository.

**Related Tutorials**

[Using Branches: git branch](https://www.atlassian.com/git/tutorials/using-branches?section=git-branch)[Using Branches: git checkout](https://www.atlassian.com/git/tutorials/using-branches?section=git-checkout)[Using Branches: git merge](https://www.atlassian.com/git/tutorials/git-merge)[Learn Git with Bitbucket Cloud: Use a Git branch to merge a file](https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud?section=git-branch-to-merge)

git checkout

In addition to checking out old commits and old file revisions, git checkout is also the means to navigate existing branches. Combined with the basic Git commands, it’s a way to work on a particular line of development.

**Related Tutorials**

[Using Branches: git checkout](https://www.atlassian.com/git/tutorials/using-branches?section=git-checkout)[Undoing Changes: git checkout](https://www.atlassian.com/git/tutorials/undoing-changes?section=git-checkout)[Comparing Workflows: Gitflow Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows?section=gitflow-workflow)

git clean

Removes untracked files from the working directory. This is the logical counterpart to git reset, which (typically) only operates on tracked files.

**Related Tutorials**

[Undoing Changes: git clean](https://www.atlassian.com/git/tutorials/undoing-changes?section=git-clean)

git clone

Creates a copy of an existing Git repository. Cloning is the most common way for developers to obtain a working copy of a central repository.

**Related Tutorials**

[Git LFS](https://www.atlassian.com/git/tutorials/git-lfs)[Comparing Workflows: Forking Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows?section=forking-workflow)[Setting up a repository: git clone](https://www.atlassian.com/git/tutorials/setting-up-a-repository?section=git-clone)

git commit

Takes the staged snapshot and commits it to the project history. Combined with git add, this defines the basic workflow for all Git users.

**Related Tutorials**

[Using Branches: git merge](https://www.atlassian.com/git/tutorials/git-merge)[Rewriting history: git commit --amend](https://www.atlassian.com/git/tutorials/rewriting-history?section=git-commit--ammend)[Learn Git with Bitbucket Cloud: Copy your Git repository and add files](https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud?section=copy-and-add-files)[Saving changes: git add](https://www.atlassian.com/git/tutorials/saving-changes?section=git-add)

git commit --amend

Passing the --amend flag to git commit lets you amend the most recent commit. This is very useful when you forget to stage a file or omit important information from the commit message.

**Related Tutorials**

[Rewriting history: git commit --amend](https://www.atlassian.com/git/tutorials/rewriting-history?section=git-commit--amend)

git config

A convenient way to set configuration options for your Git installation. You’ll typically only need to use this immediately after installing Git on a new development machine.

**Related Tutorials**

[Setting up a repository: git config](https://www.atlassian.com/git/tutorials/setting-up-a-repository?section=git-config)[Git LFS](https://www.atlassian.com/git/tutorials/git-lfs)[Install Git: Install Git on Mac OS X](https://www.atlassian.com/git/tutorials/install-git?section=mac-os-x)[Install Git: Install Git on Linux](https://www.atlassian.com/git/tutorials/install-git?section=linux)

git fetch

Fetching downloads a branch from another repository, along with all of its associated commits and files. But, it doesn't try to integrate anything into your local repository. This gives you a chance to inspect changes before merging them with your project.

**Related Tutorials**

[Syncing: git fetch](https://www.atlassian.com/git/tutorials/syncing?section=git-fetch)[Refs and the Reflog: Refspecs](https://www.atlassian.com/git/tutorials/refs-and-the-reflog?section=refspecs)[Syncing: git pull](https://www.atlassian.com/git/tutorials/syncing?section=git-pull)

git init

Initializes a new Git repository. If you want to place a project under revision control, this is the first command you need to learn.

**Related Tutorials**

[Setting up a repository: git init](https://www.atlassian.com/git/tutorials/setting-up-a-repository?section=git-init)

git log

Lets you explore the previous revisions of a project. It provides several formatting options for displaying committed snapshots.

**Related Tutorials**

[Inspecting a repository: git log](https://www.atlassian.com/git/tutorials/inspecting-a-repository?section=git-log)[Advanced Git log: Filtering the Commit History](https://www.atlassian.com/git/tutorials/git-log?section=filtering-the-commit-history)[Advanced Git log: Formatting Log Output](https://www.atlassian.com/git/tutorials/git-log?section=formatting-log-output)[Advanced Git Tutorials: Overview](https://www.atlassian.com/git/tutorials/advanced-overview)

git merge

A powerful way to integrate changes from divergent branches. After forking the project history with git branch, git merge lets you put it back together again.

**Related Tutorials**

[Merging vs. Rebasing: Workflow Walkthrough](https://www.atlassian.com/git/tutorials/merging-vs-rebasing?section=workflow-walkthrough)[Using Branches: git merge](https://www.atlassian.com/git/tutorials/git-merge)[Comparing Workflows: Gitflow Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows?section=gitflow-workflow)[Merging vs. Rebasing: Conceptual Overview](https://www.atlassian.com/git/tutorials/merging-vs-rebasing?section=conceptual-overview)

git pull

Pulling is the automated version of git fetch. It downloads a branch from a remote repository, then immediately merges it into the current branch. This is the Git equivalent of svn update.

**Related Tutorials**

[Syncing: git pull](https://www.atlassian.com/git/tutorials/syncing?section=git-pull)[Comparing Workflows: Centralized Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows?section=centralized-workflow)[Git LFS](https://www.atlassian.com/git/tutorials/git-lfs)[Comparing Workflows: Forking Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows?section=forking-workflow)

git push

Pushing is the opposite of fetching (with a few caveats). It lets you move a local branch to another repository, which serves as a convenient way to publish contributions. This is like svn commit, but it sends a series of commits instead of a single changeset.

**Related Tutorials**

[Syncing: git push](https://www.atlassian.com/git/tutorials/syncing)[Refs and the Reflog: Refspecs](https://www.atlassian.com/git/tutorials/refs-and-the-reflog?section=refspecs)[Comparing Workflows: Gitflow Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows?section=gitflow-workflow)[Git LFS](https://www.atlassian.com/git/tutorials/git-lfs)

git rebase

Rebasing lets you move branches around, which helps you avoid unnecessary merge commits. The resulting linear history is often much easier to understand and explore.

**Related Tutorials**

[Merging vs. Rebasing: Workflow Walkthrough](https://www.atlassian.com/git/tutorials/merging-vs-rebasing?section=workflow-walkthrough)[Rewriting history: git rebase -i](https://www.atlassian.com/git/tutorials/rewriting-history?section=git-rebase-i)[Merging vs. Rebasing: Conceptual Overview](https://www.atlassian.com/git/tutorials/merging-vs-rebasing?section=conceptual-overview)[Rewriting history: git rebase](https://www.atlassian.com/git/tutorials/rewriting-history?section=git-rebase)

git rebase -i

The -i flag is used to begin an interactive rebasing session. This provides all the benefits of a normal rebase, but gives you the opportunity to add, edit, or delete commits along the way.

**Related Tutorials**

[Rewriting history: git rebase -i](https://www.atlassian.com/git/tutorials/rewriting-history?section=git-rebase-i)

git reflog

Git keeps track of updates to the tip of branches using a mechanism called reflog. This allows you to go back to changesets even though they are not referenced by any branch or tag.

**Related Tutorials**

[Rewriting history: git reflog](https://www.atlassian.com/git/tutorials/rewriting-history?section=git-reflog)

git remote

A convenient tool for administering remote connections. Instead of passing the full URL to the fetch, pull, and push commands, it lets you use a more meaningful shortcut.

**Related Tutorials**

[Syncing: git remote](https://www.atlassian.com/git/tutorials/syncing?section=git-remote)

git reset

Undoes changes to files in the working directory. Resetting lets you clean up or completely remove changes that have not been pushed to a public repository.

**Related Tutorials**

[Undoing Changes: git reset](https://www.atlassian.com/git/tutorials/undoing-changes?section=git-reset)[Reset, Checkout, and Revert: Commit-level Operation](https://www.atlassian.com/git/tutorials/resetting-checking-out-and-reverting?section=commit-level-operations)[Reset, Checkout, and Revert: File-level Operations](https://www.atlassian.com/git/tutorials/resetting-checking-out-and-reverting?section=file-level-operations)[Undoing Changes: git clean](https://www.atlassian.com/git/tutorials/undoing-changes?section=git-clean)

git revert

Undoes a committed snapshot. When you discover a faulty commit, reverting is a safe and easy way to completely remove it from the code base.

**Related Tutorials**

[Undoing Changes: git revert](https://www.atlassian.com/git/tutorials/undoing-changes?section=git-revert)[Reset, Checkout, and Revert: Commit-level Operation](https://www.atlassian.com/git/tutorials/resetting-checking-out-and-reverting?section=commit-level-operation)[Reset, Checkout, and Revert: Summary](https://www.atlassian.com/git/tutorials/resetting-checking-out-and-reverting?section=summary)

git status

Displays the state of the working directory and the staged snapshot. You’ll want to run this in conjunction with git add and git commit to see exactly what’s being included in the next snapshot.

**Related Tutorials**

[Inspecting a repository: git status](https://www.atlassian.com/git/tutorials/inspecting-a-repository?section=git-status)[Git Stash](https://www.atlassian.com/git/tutorials/saving-changes/git-stash)[Learn Git with Bitbucket Cloud: Use a Git branch to merge a file](https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud?section=git-branch-to-merge)[Learn Git with Bitbucket Cloud: Copy your Git repository and add files](https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud?section=copy-and-add-files)