Command	Description	Arguments	Options	Syntax
git help	Displays help information about Git commands.	<command>	-	git help [<command>]
git version	Shows the Git version installed on the system.	-	-	git version
git clone	Creates a copy of an existing Git repository.	<repository> [<directory>]	--branch <branch>, --single-branch	git clone [<options>] <repository> [<directory>]
git clone--branch <branch>	Clone a specific branch from the repository.			git clone --branch <branch> <repository> [<directory>]
git clone--single-branch	Clone only the history leading to the tip of a single branch.			git clone --single-branch <repository> [<directory>]
git init	Initializes a new Git repository in the specified directory.	[<directory>]	--bare	git init [<directory>]
git init--bare	Create a bare repository without a working directory.			git init --bare [<directory>]
git add	Adds file changes to the staging area.	<file>...	-A, -u, -p, -i	git add [<options>] <file>...
git add-A	Stage all changes, including deletions.			git add -A <file>...
git add-u	Stage changes to already tracked files.			git add -u <file>...
git add-p	Interactively select changes to stage.			git add -p <file>...

git add-i	Interactively add changes to the index.			git add -i <file>...
git mv	Moves or renames a file, directory, or symlink.	<source> <destination>	-	git mv [<options>] <source> <destination>
git restore	Restores working directory files to their state from a commit or branch.	<file>...	--source <commit>, --staged	git restore [<options>] <file>...
git restore--source <commit>	Restore the working directory from a specific commit.			git restore --source <commit> <file>...
git restore--staged	Restore files in the staging area to match the working directory.			git restore --staged <file>...
git rm	Removes files from the working directory and stages the removal.	<file>...	--cached, --force	git rm [<options>] <file>...
git rm--cached	Remove files from the index without deleting them from the working directory.			git rm --cached <file>...
git rm--force	Forcefully remove files, even if they are modified.			git rm --force <file>...
git bisect	Uses binary search to find the commit that introduced a bug.	<command>	start, bad <commit>, good <commit>	git bisect <command> [<args>]

git bisectstart	Start a new bisect session.			git bisect start
git bisectbad <commit>	Mark a commit as bad (i.e., the bug exists in this commit).			git bisect bad <commit>
git bisectgood <commit>	Mark a commit as good (i.e., the bug does not exist in this commit).			git bisect good <commit>
git diff	Shows differences between commits, commit and working directory, etc.	[<commit>] [<commit>]	--cached, --name-only, --color	git diff [<options>] [<commit>] [<commit>]
git diff--cached	Show changes between the index and the last commit.			git diff --cached [<commit>] [<commit>]
git diff--name-only	Show only the names of changed files.			git diff --name-only [<commit>] [<commit>]
git diff--color	Show differences with color.			git diff --color [<commit>] [<commit>]
git grep	Searches for a string in the repository.	<pattern>	--cached, --ignore-case, --perl-regexp	git grep [<options>] <pattern>
git grep--cached	Search in the index (staged files).			git grep --cached <pattern>
git grep--ignore-case	Ignore case when searching.			git grep --ignore-case <pattern>
git grep--perl-regexp	Interpret the pattern as a Perl-compatible regular expression.			git grep --perl-regexp <pattern>

git log	Shows the commit logs.	[<options>]	--oneline, --graph, --author=<author>	git log [<options>]
git log--oneline	Show each commit on a single line.			git log --oneline
git log--graph	Show a graph of the commit history.			git log --graph
git log--author=<author>	Show commits by a specific author.			git log --author=<author>
git show	Shows various types of objects, including commits and tags.	[<object>]	--stat, --patch, --pretty	git show [<options>] [<object>]
git show--stat	Show statistics for the commit.			git show --stat [<object>]
git show--patch	Show the patch (diff) introduced by the commit.			git show --patch [<object>]
git show--pretty	Show commit information in a custom format.			git show --pretty=<format> [<object>]
git status	Displays the state of the working directory and staging area.	-	--short, --branch, --porcelain	git status [<options>]
git status--short	Show the status in a short format.			git status --short
git status--branch	Show the branch information in the status output.			git status --branch
git status--porcelain	Show the status in a stable format for scripts.			git status --porcelain
git branch	Lists, creates, or deletes branches.	[<branch>]	-d, -m, --list, -a	git branch [<options>] [<branch>]

git branch-d	Delete a branch.			git branch -d <branch>
git branch-m	Rename a branch.			git branch -m <old-branch> <new-branch>
git branch--list	List all branches.			git branch --list
git branch-a	List all branches, including remote branches.			git branch -a
git commit	Records changes to the repository.	-	-m <msg>, -a, --amend, -v	git commit [<options>]
git commit-m <msg>	Provide a commit message.			git commit -m <msg>
git commit-a	Automatically stage files that have been modified and deleted.			git commit -a
git commit--amend	Amend the last commit.			git commit --amend
git commit-v	Show the diff of changes in the commit message editor.			git commit -v
git merge	Joins two or more development histories together.	<branch>	--no-commit, --no-ff	git merge [<options>] <branch>
git merge--no-commit	Merge without committing automatically.			git merge --no-commit <branch>

git merge--no-ff	Merge with a merge commit even if the merge resolves as a fast-forward.			git merge --no-ff <branch>
git rebase	Reapplies commits on top of another base tip.	<branch>	--interactive, --onto, --abort	git rebase [<options>] <branch>
git rebase--interactive	Rebase commits interactively.			git rebase --interactive <branch>
git rebase--onto	Rebase commits onto another base.			git rebase --onto <new-base> <upstream> [<branch>]
git rebase--abort	Abort a rebase operation.			git rebase --abort
git reset	Resets current HEAD to the specified state.	[<commit>]	--hard, --soft, --mixed	git reset [<options>] [<commit>]
git reset--hard	Reset the working directory and index to match the commit.			git reset --hard [<commit>]
git reset--soft	Reset the index without changing the working directory.			git reset --soft [<commit>]
git reset--mixed	Reset the index and update the working directory to match the commit.			git reset --mixed [<commit>]
git switch	Switches branches or restores working tree files.	<branch>	-c, --discard-changes	git switch [<options>] <branch>

git switch-c	Create and switch to a new branch.			git switch -c <branch>
git switch--discard-changes	Discard changes in the working directory when switching branches.			git switch --discard-changes <branch>
git tag	Creates, lists, or deletes tags.	[<tag>]	-d, -a, -s, -f	git tag [<options>] [<tag>]
git tag-d	Delete a tag.			git tag -d <tag>
git tag-a	Create an annotated tag.			git tag -a <tag> -m <message>
git tag-s	Create a signed tag.			git tag -s <tag> -m <message>
git tag-f	Force tag creation, overwriting any existing tag with the same name.			git tag -f <tag>
git fetch	Downloads objects and refs from another repository.	[<remote>]	--all, --prune, --tags	git fetch [<options>] [<remote>]
git fetch--all	Fetch all remotes.			git fetch --all
git fetch--prune	Remove stale remote-tracking branches.			git fetch --prune
git fetch--tags	Fetch all tags from the remote repository.			git fetch --tags
git pull	Fetches and integrates with another repository or a local branch.	[<remote>] [<branch>]	--rebase, --no-ff, --all	git pull [<options>] [<remote>] [<branch>]

git pull--rebase	Rebase the current branch on top of the upstream branch after fetching.			git pull --rebase [<remote>] [<branch>]
git pull--no-ff	Perform a merge commit even if a fast-forward is possible.			git pull --no-ff [<remote>] [<branch>]
git push	Updates remote refs along with associated objects.	[<remote>] [<branch>]	--force, --set-upstream, --all	git push [<options>] [<remote>] [<branch>]
git push--force	Forcefully push changes to the remote repository.			git push --force [<remote>] [<branch>]
git push--set-upstream	Set the upstream branch for the current branch.			git push --set-upstream [<remote>] [<branch>]
git push--all	Push all branches to the remote repository.			git push --all [<remote>]
git config	Gets and sets configuration variables.	[<key> [<value>]]	--global, --system, --local	git config [<options>] [<key> [<value>]]
git config--global	Set configuration options globally.			git config --global <key> [<value>]
git config--system	Set configuration options system-wide.			git config --system <key> [<value>]
git config--local	Set configuration options for the current repository.			git config --local <key> [<value>]

git am	Applies patches from mailbox.	[<file>]	--3way, --signoff, --skip	git am [<options>] [<file>]
git am--3way	Use three-way merge to apply patches.			git am --3way [<file>]
git am--signoff	Add a "Signed-off-by" line to the commit message.			git am --signoff [<file>]
git am--skip	Skip applying the patch if it fails.			git am --skip [<file>]
git archive	Creates an archive of files from a named tree.	<tree-ish> [<path>]	--format=<format>, --output=<file>	git archive [<options>] <tree-ish> [<path>]
git archive--format=<format>	Specify the format of the archive (e.g., tar, zip).			git archive --format=<format> <tree-ish> [<path>]
git archive--output=<file>	Write the archive to a file.			git archive --output=<file> <tree-ish> [<path>]
git bundle	Creates and manages bundles of Git objects.	<create	verify	unbundle>
git bundlecreate <file>	Create a new bundle file.			git bundle create <file> <commit>...
git bundleverify <file>	Verify the integrity of a bundle file.			git bundle verify <file>
git bundleunbundle <file>	Unbundle objects from a bundle file.			git bundle unbundle <file>
git checkout	Switches branches or restores working directory files.	[<branch>] [<file>]	-b, --orphan, --track	`git checkout [<options>] [<branch>
git checkout-b	Create and switch to a new branch.			git checkout -b <branch>

git checkout--orphan	Create a new orphan branch (a branch without any commit history).			git checkout --orphan <branch>
git checkout--track	Set up tracking for a new branch.			git checkout --track <remote>/<branch>
git cherry-pick	Applies the changes introduced by some existing commits.	<commit>	--no-commit, --edit, --strategy	git cherry-pick [<options>] <commit>
git cherry-pick--no-commit	Apply changes from a commit without committing them.			git cherry-pick --no-commit <commit>
git cherry-pick--edit	Edit the commit message before applying.			git cherry-pick --edit <commit>
git cherry-pick--strategy	Use a specific merge strategy when applying the commit.			git cherry-pick --strategy=<strategy> <commit>
git citool	Launches a graphical commit tool.	-	-	git citool
git clean	Removes untracked files from the working directory.	-	-f, -d, -x	git clean [<options>]
git clean-f	Forcefully remove untracked files.			git clean -f
git clean-d	Remove untracked directories.			git clean -d
git clean-x	Remove all untracked files, including ignored files.			git clean -x

git describe	Describes the current commit using the most recent tag reachable from it.	[<commit>]	--tags, --always, --abbrev=<n>	git describe [<options>] [<commit>]
git describe--tags	Describe the commit using the nearest tag.			git describe --tags [<commit>]
git describe--always	Always show the unique hash if no tags are found.			git describe --always [<commit>]
git describe--abbrev=<n>	Set the length of the abbreviated object name.			git describe --abbrev=<n> [<commit>]
git format-patch	Prepares patches for email submission.	[<range>]	--stdout, --cover-letter	git format-patch [<options>] [<range>]
git format-patch--stdout	Output patches to the standard output instead of files.			git format-patch --stdout [<range>]
git format-patch--cover-letter	Create a cover letter for the patches.			git format-patch --cover-letter [<range>]
git gc	Optimizes the repository by cleaning up unnecessary files and optimizing the local repository.	-	--auto, --aggressive, --prune	git gc [<options>]
git gc--auto	Automatically optimize the repository.			git gc --auto

git gc--aggressive	Perform more thorough optimization.			git gc --aggressive
git gc--prune	Remove objects older than a specified time.			git gc --prune=<time>
git core-tutorial	Provides a tutorial for Git core concepts.	-	-	git core-tutorial
git credentials	Manages credentials for Git operations.	-	--help, --list, --store, --erase	git credentials [<options>]
git credentials--help	Show help information about credential management.			git credentials --help
git credentials--list	List all stored credentials.			git credentials --list
git credentials--store	Store credentials.			git credentials --store
git credentials--erase	Erase stored credentials.			git credentials --erase
git cvs-migration	Facilitates migration from CVS to Git.	-	-	git cvs-migration
git diffcore	Provides details about the diff algorithm.	-	-	git diffcore
git everyday	Provides a guide to everyday Git usage.	-	-	git everyday
git faq	Provides frequently asked questions about Git.	-	-	git faq
git glossary	Provides a glossary of Git terms.	-	-	git glossary

git namespaces	Provides information about Git namespaces.	-	-	git namespaces
git remote-helpers	Manages remote helpers for various protocols.	-	-	git remote-helpers
git submodules	Manages Git submodules.	[<command>]	--recursive, --quiet	git submodule [<options>] <command>
git submodule--recursive	Initialize and update nested submodules.			git submodule --recursive <command>
git submodule--quiet	Suppress output.			git submodule --quiet <command>
git tutorial	Provides an introductory tutorial for Git.	-	-	git tutorial
git tutorial-2	Provides an advanced Git tutorial.	-	-	git tutorial-2
git workflows	Provides information on various Git workflows.	-	-	git workflows
git bugreport	Provides a way to report Git bugs.	-	-	git bugreport
git Credential helpers	Manages credential helpers for Git.	-	-	git credential [<options>]
git notes	Manage and view notes added to objects.		add, edit, list, remove, show	git notes <command> [options]
git notes add	Add notes to objects.			git notes add [options] [<object>]
git notes edit	Edit existing notes.			git notes edit [options] [<object>]

git notes list	List all notes.			git notes list [options]
git notes remove	Remove notes from objects.			git notes remove [options] [<object>]
git notes show	Show notes attached to an object.			git notes show [options] [<object>]
git mergetool	Launch a merge tool to resolve merge conflicts.		--tool=<tool>, --no-prompt, --tool-help	git mergetool [options]
git mergetool --tool=<tool>	Specify the merge tool to use.			git mergetool --tool=<tool> [options]
git mergetool --no-prompt	Disable prompting for each file.			git mergetool --no-prompt [options]
git mergetool --tool-help	Display available merge tools.			git mergetool --tool-help [options]
git stash	Stash the changes in a dirty working directory away.		push, pop, apply, list, show, drop, clear	git stash <command> [options]
git stash push	Stash the changes in the working directory.			git stash push [options]
git stash pop	Apply the most recent stash and remove it from the stash list.			git stash pop [options]
git stash apply	Apply a stash to the working directory.			git stash apply [options] [<stash>]
git stash list	List all stashes.			git stash list [options]
git stash show	Show the changes in a stash.			git stash show [options] [<stash>]

git stash drop	Remove a stash from the stash list.			git stash drop [options] [<stash>]
git stash clear	Remove all stashes.			git stash clear [options]
git worktree	Manage multiple working trees attached to the same repository.		add, list, remove, prune	git worktree <command> [options]
git worktree add	Add a new working tree.			git worktree add [options] <path> [<branch>]
git worktree list	List all working trees.			git worktree list [options]
git worktree remove	Remove a working tree.			git worktree remove [options] <path>
git worktree prune	Remove stale working trees.			git worktree prune [options]
git remote	Manage set of tracked repositories.		add, remove, rename, set-url, show, prune	git remote <command> [options]
git remote add	Add a new remote repository.			git remote add [options] <name> <url>
git remote remove	Remove a remote repository.			git remote remove [options] <name>
git remote rename	Rename a remote repository.			git remote rename [options] <old-name> <new-name>
git remote set-url	Change the URL of a remote repository.			git remote set-url [options] <name> <url>

git remote show	Display information about a remote.			git remote show [options] [<name>]
git remote prune	Prune obsolete remote-tracking references.			git remote prune [options] <name>
git submodule	Initialize, update, or inspect submodules.		add, status, update, init, sync, foreach	git submodule <command> [options]
git submodule add	Add a new submodule.			git submodule add [options] <repository> [<path>]
git submodule status	Show the status of submodules.			git submodule status [options]
git submodule update	Update submodules.			git submodule update [options]
git submodule init	Initialize submodules.			git submodule init [options]
git submodule sync	Synchronize submodule URLs.			git submodule sync [options]
git submodule foreach	Execute a command in each submodule.			git submodule foreach [options] <command>
git difftool	Launch a diff tool to compare changes.		--tool=<tool>, --no-prompt, --tool-help	git difftool [options]
git difftool--tool=<tool>	Specify the diff tool to use.			git difftool --tool=<tool> [options]
git difftool--no-prompt	Disable prompting for each file.			git difftool --no-prompt [options]
git difftool--tool-help	Display available diff tools.			git difftool --tool-help [options]

git range-diff	Show the difference between two ranges of commits.		--range=<range>	git range-diff <range1> <range2>
git shortlog	Summarize git log output.		-s, -n, -e, --no-merges, --summary	git shortlog [options] [<revision range>]
git shortlog -s	Show a summary of commits per author.			git shortlog -s [options]
git shortlog -n	Sort by the number of commits.			git shortlog -n [options]
git shortlog -e	Show email addresses for each author.			git shortlog -e [options]
git shortlog--no-merges	Exclude merge commits.			git shortlog --no-merges [options]
git shortlog--summary	Show a summary of the commits.			git shortlog --summary [options]
git apply	Apply a patch to files and/or index.		--index, --cached, --reject, --ignore-space-change	git apply [options] <patch-file>
git apply--index	Apply changes to the index (staging area).			git apply --index [options] <patch-file>
git apply--cached	Apply changes only to the index.			git apply --cached [options] <patch-file>
git apply--reject	Save rejected patches in .rej files.			git apply --reject [options] <patch-file>
git apply--ignore-space-change	Ignore changes in the amount of white space.			git apply --ignore-space-change [options] <patch-file>

git revert	Revert changes by creating a new commit that undoes the changes.		-n, --no-edit, --no-commit, --signoff	git revert [options] <commit>
git revert-n	Perform a revert but do not commit.			git revert -n [options] <commit>
git revert--no-edit	Do not open an editor to modify the commit message.			git revert --no-edit [options] <commit>
git revert--no-commit	Apply the changes but do not commit them.			git revert --no-commit [options] <commit>
git revert--signoff	Add a "Signed-off-by" line to the commit message.			git revert --signoff [options] <commit>
git blame	Show what revision and author last modified each line of a file.		-L <start>,<end>, --show-number, --incremental, --porcelain	git blame [options] <file>
git blame-L <start>,<end>	Annotate the lines between start and end.			git blame -L <start>,<end> [options] <file>
git blame--show-number	Show line numbers in the output.			git blame --show-number [options] <file>
git blame--incremental	Output in incremental mode.			git blame --incremental [options] <file>
git blame--porcelain	Produce machine-readable output.			git blame --porcelain [options] <file>
git attributes	Show or set attributes for files in the repository.		--get, --set, --unset, --list, --show	git gitattributes <command> [options]

git attributes--get	Get the value of an attribute.			git gitattributes --get [options] <file>
git attributes--set	Set the value of an attribute.			git gitattributes --set [options] <file>
git attributes--unset	Unset the value of an attribute.			git gitattributes --unset [options] <file>
git attributes--list	List all attributes.			git gitattributes --list [options]
git attributes--show	Show the attributes for a file.			git gitattributes --show [options] <file>
git Command-line interface conventions	Overview of the conventions for Git command-line options and arguments.		-	-
git Everyday Git	Practical usage tips for daily Git operations.		-	-
git Frequently Asked Questions (FAQ)	Common questions and answers about Git.		-	-
git Hooks	Scripts that Git executes before or after events such as commit, push, and receive.		pre-commit, post-commit, pre-push, post-receive	git <hook-name>
git Hooks pre-commit	Runs before a commit is made. Commonly used to perform tasks like linting or running tests.			.git/hooks/pre-commit

git Hooks post-commit	Runs after a commit is made. Often used for notifications or additional processes.			.git/hooks/post-commit
git Hooks pre-push	Runs before a push is made to a remote repository. Used for tasks like checking the code before pushing.			.git/hooks/pre-push
git Hooks post-receive	Runs after a push is received by the remote repository. Used for tasks like deployment or notifications.			.git/hooks/post-receive
git ignore	Specifies files and directories that Git should ignore.		--get, --set, --unset	git gitignore [options] <patterns>
git ignore--get	Get the value of a .gitignore pattern.			git gitignore --get [options] <patterns>
git ignore--set	Set a .gitignore pattern.			git gitignore --set [options] <patterns>
git ignore--unset	Remove a .gitignore pattern.			git gitignore --unset [options] <patterns>
git modules	Manage the repository's configuration for submodules.		--get, --set, --unset	git gitmodules <command> [options]

git modules--get	Get the URL or path of a submodule.			git gitmodules --get [options] <submodule>
git modules--set	Set the URL or path of a submodule.			git gitmodules --set [options] <submodule>
git modules--unset	Remove the URL or path of a submodule.			git gitmodules --unset [options] <submodule>
git Revisions	Various ways to specify revisions in Git.		HEAD, HEAD~1, branch-name, commit-hash, @{<date>}, etc.	-
git Revisions HEAD	Points to the latest commit on the current branch.			git <command> HEAD
git Revisions HEAD~1	Refers to the commit before HEAD.			git <command> HEAD~1
git Revisions branch-name	Refers to a branch's latest commit.			git <command> branch-name
git Revisions commit-hash	Refers to a specific commit by its hash.			git <command> <commit-hash>
git Revisions @{<date>}	Refers to the commit at a specific date.			git <command> @{<date>}
git send-email	Send an email with patches attached.		--to=<address>, --cc=<address>, --subject=<text>, --smtp-server=<host>	git send-email [options] <path-to-patch>
git send-email--to=<address>	Specify recipient email address.			git send-email --to=<address> [options] <path-to-patch>

git send-email--cc=<address>	Specify CC email addresses.			git send-email --cc=<address> [options] <path-to-patch>
git send-email--subject=<text>	Specify the email subject.			git send-email --subject=<text> [options] <path-to-patch>
git send-email--smtp-server=<host>	Specify the SMTP server to use for sending emails.			git send-email --smtp-server=<host> [options] <path-to-patch>
git request-pull	Request a pull from a repository.		--no-edit, --subject=<text>, --to=<address>	git request-pull [options] <start> <url> <end>
git request-pull--no-edit	Do not edit the pull request message.			git request-pull --no-edit [options] <start> <url> <end>
git request-pull--subject=<text>	Specify the subject of the pull request email.			git request-pull --subject=<text> [options] <start> <url> <end>
git request-pull--to=<address>	Specify recipient email address for the pull request.			git request-pull --to=<address> [options] <start> <url> <end>
git svn	Bidirectional operation between Git and Subversion repositories.		clone, rebase, dcommit, fetch, log, info	git svn <command> [options]

git svn clone	Clone a Subversion repository into a Git repository.			git svn clone [options] <url>
git svn rebase	Rebase your changes on top of the latest upstream changes.			git svn rebase [options]
git svn dcommit	Commit changes from Git to the Subversion repository.			git svn dcommit [options]
git svn fetch	Fetch changes from the Subversion repository.			git svn fetch [options]
git svn log	Show commit logs from the Subversion repository.			git svn log [options]
git svn info	Display information about the Subversion repository.			git svn info [options]
git fast-import	Import data into a Git repository from another format.		-	git fast-import [options]
git fsck	Verify the connectivity and validity of objects in the repository.		--full, --strict, --lost-found	git fsck [options]
git fsck--full	Perform a full check of the repository.			git fsck --full [options]
git fsck--strict	Perform strict integrity checking.			git fsck --strict [options]

git fsck--lost-found	Write out missing or corrupted objects.			git fsck --lost-found [options]
git reflog	Show the reference logs.		--decorate, --format=<format>, --no-abbrev	git reflog [options]
git reflog--decorate	Show references and associated logs.			git reflog --decorate [options]
git reflog--format=<format>	Specify the format of the output.			git reflog --format=<format> [options]
git reflog--no-abbrev	Show full commit hashes in the output.			git reflog --no-abbrev [options]
git filter-branch	Rewrite branches and tags with a new history.		--tree-filter=<cmd>, --index-filter=<cmd>, --msg-filter=<cmd>, --commit-filter=<cmd>	git filter-branch [options] <refspec>
git filter-branch--tree-filter=<cmd>	Run a command to modify the tree during filtering.			git filter-branch --tree-filter=<cmd> [options]
git filter-branch--index-filter=<cmd>	Run a command to modify the index during filtering.			git filter-branch --index-filter=<cmd> [options]
git filter-branch--msg-filter=<cmd>	Run a command to modify commit messages during filtering.			git filter-branch --msg-filter=<cmd> [options]
git filter-branch--commit-filter=<cmd>	Run a command to modify commits during filtering.			git filter-branch --commit-filter=<cmd> [options]

git instaweb	Instantly browse your repository with a web browser.		--httpd=<server>, --local, --browse, --port=<port>	git instaweb [options]
git instaweb--httpd=<server>	Specify the HTTP server to use.			git instaweb --httpd=<server> [options]
git instaweb--local	Start the web server in local mode.			git instaweb --local [options]
git instaweb--browse	Open the repository in a web browser.			git instaweb --browse [options]
git instaweb--port=<port>	Specify the port for the web server.			git instaweb --port=<port> [options]
git archive	Create an archive of files from a named tree.		--format=<format>, --output=<file>, --prefix=<prefix>	git archive [options] <tree-ish>
git archive--format=<format>	Specify the archive format (e.g., tar, zip).			git archive --format=<format> [options] <tree-ish>
git archive--output=<file>	Write the archive to a file.			git archive --output=<file> [options] <tree-ish>
git archive--prefix=<prefix>	Prefix file names in the archive.			git archive --prefix=<prefix> [options] <tree-ish>
git daemon	A simple server for Git repositories.		--base-path=<dir>, --export-all, --detach	git daemon [options]
git daemon--base-path=<dir>	Specify the base path for the repositories.			git daemon --base-path=<dir> [options]

git daemon--export-all	Export all repositories in the base path.			git daemon --export-all [options]
git daemon--detach	Run the daemon in the background.			git daemon --detach [options]
git update-server-info	Update server info files for repositories.		-	git update-server-info
git cat-file	Provide content or type of objects.		-t, -s, -e, -p	git cat-file [options] <object>
git cat-file-t	Show the type of an object.			git cat-file -t <object>
git cat-file-s	Show the size of an object.			git cat-file -s <object>
git cat-file-e	Check if an object exists.			git cat-file -e <object>
git cat-file-p	Show the contents of an object.			git cat-file -p <object>
git check-ignore	Check if paths are ignored by the .gitignore file.		-v, --no-index	git check-ignore [options] <path>
git check-ignore-v	Show which pattern is ignoring the file.			git check-ignore -v <path>
git check-ignore--no-index	Check ignore status outside of a Git repository.			git check-ignore --no-index <path>
git checkout-index	Checkout files from the index.		-a, -f, --prefix=<prefix>	git checkout-index [options]
git checkout-index-a	Checkout all files from the index.			git checkout-index -a [options]
git checkout-index-f	Force checkout of files.			git checkout-index -f [options]

git checkout-index--prefix=<prefix>	Apply a prefix to file names.			git checkout-index --prefix=<prefix> [options]
git commit-tree	Create a commit object.		-p, -m, -a, --author=<name>	git commit-tree <tree> [options]
git commit-tree-p	Parent commit(s) for the new commit.			git commit-tree -p <parent> [options] <tree>
git commit-tree-m	Commit message.			git commit-tree -m <message> [options] <tree>
git commit-tree-a	Automatically include all changes.			git commit-tree -a <tree>
git commit-tree--author=<name>	Specify the author of the commit.			git commit-tree --author=<name> [options] <tree>
git count-objects	Count the number of objects in the database.		-v, -H, -z	git count-objects [options]
git count-objects-v	Show verbose output including size and count of objects.			git count-objects -v [options]
git count-objects-H	Display human-readable output.			git count-objects -H [options]
git count-objects-z	Output in a format suitable for scripts.			git count-objects -z [options]
git diff-index	Show changes between the index and a tree.		--cached, --name-only, --name-status, --diff-filter=<filter>	git diff-index [options] <tree>
git diff-index--cached	Compare the index against the tree.			git diff-index --cached [options] <tree-ish>

git diff-index--name-only	Show only file names that differ.			git diff-index --name-only [options] <tree-ish>
git diff-index--name-status	Show file names and their status.			git diff-index --name-status [options] <tree-ish>
git diff-index--diff-filter=<filter>	Show only files that match the filter.			git diff-index --diff-filter=<filter> [options] <tree-ish>
git for-each-ref	Iterate over all refs in the repository.		--format=<format>, --sort=<key>, --contains=<commit>, --points-at=<ref>	git for-each-ref [options] [<ref-pattern>]
git for-each-ref--format=<format>	Specify the output format for each reference.			git for-each-ref --format=<format> [options] <ref>
git for-each-ref--sort=<key>	Sort references by key.			git for-each-ref --sort=<key> [options] <ref>
git for-each-ref--contains=<commit>	Show only references that contain a specific commit.			git for-each-ref --contains=<commit> [options] <ref>
git hash-object	Compute the object ID of a file.		-w, --stdin, --literally	git hash-object [options] <file>
git hash-object-w	Write the object to the object database.			git hash-object -w <file>
git hash-object--stdin	Read file names from standard input.			git hash-object --stdin
git hash-object--literally	Read objects as they are, without filtering.			git hash-object --literally <file>

git ls-files	Show information about files in the index and working directory.		--stage, --cached, --deleted, --ignored	git ls-files [options]
git ls-files--stage	Show the staged contents of the index.			git ls-files --stage [options]
git ls-files--cached	Show files in the index.			git ls-files --cached [options]
git ls-files--deleted	Show files that are deleted.			git ls-files --deleted [options]
git ls-files--ignored	Show ignored files.			git ls-files --ignored [options]
git ls-tree	List the contents of a tree object.		-r, -t, -d, -l	git ls-tree [options] <tree>
git ls-tree-r	Recursively list contents.			git ls-tree -r [options] <tree>
git ls-tree-t	List only tree objects.			git ls-tree -t [options] <tree>
git ls-tree-d	List only directories.			git ls-tree -d [options] <tree>
git ls-tree-l	Show object information.			git ls-tree -l [options] <tree>
git merge-base	Find the common ancestor of two commits.		--fork-point, --octopus, --independent, --all	git merge-base [options] <commit1> <commit2>
git merge-base--fork-point	Find the fork point for two commits.			git merge-base --fork-point [options] <commit1> <commit2>

git merge-base--octopus	Find the merge base of more than two commits.			git merge-base --octopus [options] <commit1> <commit2>
git merge-base--independent	Find merge bases that are independent of the commits.			git merge-base --independent [options] <commit1> <commit2>
git merge-base--all	Find all merge bases for multiple commits.			git merge-base --all [options] <commit1> <commit2>
git read-tree	Read tree information into the index.		--prefix=<prefix>, --empty, --index	git read-tree [options] <tree>
git read-tree--prefix=<prefix>	Apply a prefix to the pathnames in the index.			git read-tree --prefix=<prefix> [options] <tree>
git read-tree--empty	Create an empty index.			git read-tree --empty [options] <tree>
git read-tree--index	Update the index without updating the working directory.			git read-tree --index [options] <tree>
git rev-list	List commit objects in reverse chronological order.		--max-count=<n>, --skip=<n>, --since=<date>, --until=<date>	git rev-list [options] <revision>
git rev-list--max-count=<n>	Limit the number of commits to output.			git rev-list --max-count=<n> [options] <revision>
git rev-list--skip=<n>	Skip the first n commits.			git rev-list --skip=<n> [options] <revision>

git rev-list--since=<date>	Show commits more recent than a specific date.			git rev-list --since=<date> [options] <revision>
git rev-list--until=<date>	Show commits older than a specific date.			git rev-list --until=<date> [options] <revision>
git rev-parse	Parse and handle git revision and option parameters.		--abbrev-ref, --verify, --show-toplevel, --git-dir, --parseopt	git rev-parse [options] <revision>
git rev-parse--abbrev-ref	Show the abbreviated ref name.			git rev-parse --abbrev-ref [options] <revision>
git rev-parse--verify	Check if a revision exists.			git rev-parse --verify [options] <revision>
git rev-parse--show-toplevel	Show the top-level directory of the working tree.			git rev-parse --show-toplevel [options]
git rev-parse--git-dir	Show the Git directory.			git rev-parse --git-dir [options]
git rev-parse--parseopt	Parse options and parameters.			git rev-parse --parseopt [options]
git show-ref	Show references.		--heads, --tags, --dereference	git show-ref [options] [<ref>]
git show-ref--heads	Show reference for branches.			git show-ref --heads [options]
git show-ref--tags	Show reference for tags.			git show-ref --tags [options]
git show-ref--dereference	Show dereferenced references.			git show-ref --dereference [options]
git symbolic-ref	Read and update symbolic references.		--short, --delete, --create	git symbolic-ref [options] <ref>

git symbolic-ref--short	Show short ref names.			git symbolic-ref --short [options] <ref>
git symbolic-ref--delete	Delete a symbolic reference.			git symbolic-ref --delete [options] <ref>
git symbolic-ref--create	Create a symbolic reference.			git symbolic-ref --create [options] <ref>
git update-index	Register file contents in the index.		--add, --remove, --chmod=<mode>, --skip-worktree, --no-skip-worktree	git update-index [options] <file>
git update-index--add	Add a file to the index.			git update-index --add [options] <file>
git update-index--remove	Remove a file from the index.			git update-index --remove [options] <file>
git update-index--chmod=<mode>	Change file mode.			git update-index --chmod=<mode> [options] <file>
git update-index--skip-worktree	Mark a file as "skip-worktree".			git update-index --skip-worktree [options] <file>
git update-index--no-skip-worktree	Unmark a file as "skip-worktree".			git update-index --no-skip-worktree [options] <file>
git update-ref	Update the value of a reference.		--create, --delete, --force, --no-deref	git update-ref [options] <ref> <new-value> [<old-value>]
git update-ref--create	Create a new reference.			git update-ref --create [options] <ref> <new-value>

git update-ref--delete	Delete a reference.			git update-ref --delete [options] <ref>
git update-ref--force	Force update a reference.			git update-ref --force [options] <ref> <new-value>
git update-ref--no-deref	Update the reference without dereferencing.			git update-ref --no-deref [options] <ref> <new-value>
git verify-pack	Check the integrity of objects in a pack.		-v, --full, --quiet	git verify-pack [options] <packfile>
git verify-pack-v	Verbose output showing object details.			git verify-pack -v [options] <packfile>
git verify-pack--full	Perform a full check on the packfile.			git verify-pack --full [options] <packfile>
git verify-pack--quiet	Suppress non-critical messages.			git verify-pack --quiet [options] <packfile>
git write-tree	Writes the current index to the object database and returns the tree object name.	None	None	git write-tree
git cherry-pick	Apply the changes introduced by some existing commits.	<commit>	-e, --edit (edit commit message), -n, --no-commit (apply changes without committing)	git cherry-pick [options] <commit>
git cherry-pick -e, --edit	Edit the commit message.			git cherry-pick -e <commit>

git cherry-pick -n, --no-commit	Apply changes without committing.			git cherry-pick -n <commit>
git format-patch	Prepare patches for email submission.	[<options>]	-o <directory>, --stdout, --cover-letter, --numbered, -M (detect renames), etc.	git format-patch [options] <range>
git format-patch-o <directory>	Output patches to a directory.			git format-patch -o <directory> <range>
git format-patch--stdout	Output patches to stdout.			git format-patch --stdout <range>
git format-patch--cover-letter	Create a cover letter for the patches.			git format-patch --cover-letter <range>
git format-patch--numbered	Number the patches.			git format-patch --numbered <range>
git format-patch-M	Detect renames.			git format-patch -M <range>
git gui	Launches the Git GUI interface.	None	None	git gui
git maintenance	Run maintenance tasks on a Git repository.	<subcommand>	--auto, --schedule, --run, etc.	git maintenance <subcommand> [options]
git maintenance--auto	Run maintenance tasks automatically.			git maintenance --auto
git maintenance--schedule	Schedule a maintenance task.			git maintenance --schedule <task>
git maintenance--run	Run the specified maintenance task.			git maintenance --run <task>
git range-diff	Compare two sets of commits, showing the differences between them.	<base> <other>	None	git range-diff <base> <other>

git sparse-checkout	Allows partial checkout of files from a repository.	<subcommand>	--cone, --no-cone, --set, --add, --reapply	git sparse-checkout <subcommand> [options]
git sparse-checkout--cone	Use cone mode for sparse checkout.			git sparse-checkout --cone
git sparse-checkout--no-cone	Disable cone mode.			git sparse-checkout --no-cone
git sparse-checkout--set	Set the sparse-checkout configuration.			git sparse-checkout --set <patterns>
git sparse-checkout--add	Add patterns to sparse-checkout.			git sparse-checkout --add <patterns>
git sparse-checkout--reapply	Reapply sparse-checkout after a change.			git sparse-checkout --reapply
gitk	Launches a graphical history viewer for Git repositories.	None	--all, --tags, --branches, --no-graph, etc.	gitk [options]
gitk--all	Show all branches.			gitk --all
gitk--tags	Show tags.			gitk --tags
gitk--branches	Show branches.			gitk --branches
gitk--no-graph	Disable the graphical history view.			gitk --no-graph
scalar	A tool to simplify the management of large repositories.	None	None	scalar <subcommand>
git fast-export	Export the Git repository as a fast-export stream.	None	None	git fast-export [options]
git fast-import	Import a fast-import stream into a Git repository.	None	None	git fast-import [options]

git filter-branch	Rewrite branches to modify history.	None	--env-filter, --tree-filter, --index-filter, --commit-filter, etc.	git filter-branch [options] <filter>
git filter-branch--env-filter	Filter commits based on environment variables.			git filter-branch --env-filter '<script>'
git filter-branch--tree-filter	Filter tree objects.			git filter-branch --tree-filter '<script>'
git filter-branch--index-filter	Filter the index (staging area).			git filter-branch --index-filter '<script>'
git filter-branch--commit-filter	Filter commit objects.			git filter-branch --commit-filter '<script>'
git pack-refs	Pack all refs into a single file to optimize performance.	None	None	git pack-refs [options]
git prune	Remove unreachable objects from the object database.	None	--expire <time>, --dry-run	git prune [options]
git prune--expire <time>	Remove objects older than a specified time.			git prune --expire <time>
git prune--dry-run	Show which objects would be pruned without actually removing them.			git prune --dry-run
git repack	Repack all or some of the repository's object files.	None	-a, --all, -d, --delete, -l, --local	git repack [options]
git repack-a, --all	Repack all objects.			git repack -a
git repack-d, --delete	Remove redundant pack files.			git repack -d

git repack-l, --local	Optimize local repository.			git repack -l
git replace	Create, list, or delete object replacements.	<command>	--list, --delete, --stdin, --force, etc.	git replace [options] <command>
git replace--list	List all replacements.			git replace --list
git replace--delete	Delete a replacement.			git replace --delete <object>
git replace--stdin	Read replacements from stdin.			git replace --stdin
git replace--force	Force replace operation.			git replace --force <old> <new>
git annotate	Show the commit information for each line in a file.	<file>	-p, --porcelain, -a, --all, etc.	git annotate [options] <file>
git annotate-p, --porcelain	Output in a machine-readable format.			git annotate -p <file>
git annotate-a, --all	Show annotations for all lines.			git annotate -a <file>
git count-objects	Count the number of objects in the repository and their disk usage.	None	-v, --verbose, -H, --human-readable	git count-objects [options]
git count-objects-v, --verbose	Show detailed information.			git count-objects -v
git count-objects-H, --human-readable	Show sizes in human-readable format.			git count-objects -H
git diagnose	Diagnose issues with the Git repository.	None	None	git diagnose
git merge-tree	Show the merge result of three trees.	<tree1> <tree2> <tree3>	None	git merge-tree <tree1> <tree2> <tree3>

git rerere	Record and replay resolved conflicts.	<subcommand>	--edit, --clear, --verbose, etc.	git rerere <subcommand> [options]
git rerere--edit	Edit the recorded resolution.			git rerere --edit
git rerere--clear	Clear all recorded resolutions.			git rerere --clear
git rerere--verbose	Show more detailed output.			git rerere --verbose
git show-branch	Show branches and their commits.	[<branch>]	--sha1-name, --topics, --all, etc.	git show-branch [options] [<branch>]
git show-branch--sha1-name	Show SHA-1 hashes of commits.			git show-branch --sha1-name
git show-branch--topics	Show topics.			git show-branch --topics
git show-branch--all	Show all branches.			git show-branch --all
git verify-commit	Verify the integrity of a commit object.	<commit>	None	git verify-commit <commit>
git verify-tag	Verify the integrity of a tag object.	<tag>	None	git verify-tag <tag>
git whatchanged	Show changes over time, including commits and diffs.	None	-p, --patch, --no-patch, --since <date>, etc.	git whatchanged [options]
git whatchanged-p, --patch	Show the patch for each commit.			git whatchanged -p
git whatchanged--no-patch	Do not show patches.			git whatchanged --no-patch
git whatchanged--since <date>	Show commits since a specific date.			git whatchanged --since <date>
gitweb	Web-based interface to browse a Git repository.	None	None	gitweb

git archimport	Import a set of changes from an Arch repository.	None	None	git archimport
git cvsexportcommit	Export a commit to CVS.	None	None	git cvsexportcommit
git cvsimport	Import CVS repository into Git.	None	--branch, --tag, --merge, --authors-file	git cvsimport [options] <repository>
git cvsimport--branch	Import a branch from CVS.			git cvsimport --branch <branch>
git cvsimport--tag	Import a tag from CVS.			git cvsimport --tag <tag>
git cvsimport--merge	Merge CVS branches into Git branches.			git cvsimport --merge
git cvsimport--authors-file	Specify a file mapping CVS authors to Git authors.			git cvsimport --authors-file <file>
git cvsserver	Run a CVS server on a Git repository.	None	None	git cvsserver
git imap-send	Send Git commits via IMAP.	None	None	git imap-send [options]
git p4	Interface with Perforce.	<subcommand>	--port, --user, --password, etc.	git p4 <subcommand> [options]
git p4--port	Specify Perforce server port.			git p4 --port <port>
git p4--user	Specify Perforce user.			git p4 --user <user>
git p4--password	Specify Perforce password.			git p4 --password <password>
git quiltimport	Import patches created by Quilt.	None	--strip, --no-commit, etc.	git quiltimport [options] <patch>

git quiltimport--strip	Remove leading directories from filenames in the patch.			git quiltimport --strip <number>
git quiltimport--no-commit	Do not commit the imported patches automatically.			git quiltimport --no-commit <patch>
git request-pull	Generate a request for a pull from another repository.	<start> <url> <end>	--draft, --no-verify, etc.	git request-pull [options] <start> <url> <end>
git request-pull--draft	Create a draft pull request.			git request-pull --draft <start> <url> <end>
git request-pull--no-verify	Skip verification checks.			git request-pull --no-verify <start> <url> <end>
git send-email	Send commits via email.	None	--to, --cc, --bcc, --subject, --compose, etc.	git send-email [options] <files>
git send-email--to	Specify the recipients of the email.			git send-email --to <email>
git send-email--cc	Specify additional recipients to CC.			git send-email --cc <email>
git send-email--bcc	Specify additional recipients to BCC.			git send-email --bcc <email>
git send-email--subject	Specify the subject of the email.			git send-email --subject <subject>
git send-email--compose	Compose an email manually.			git send-email --compose <files>

git revert	Revert changes introduced by a commit.	<commit>	-n, --no-commit, --no-edit, etc.	git revert [options] <commit>
git revert-n, --no-commit	Apply changes without committing.			git revert -n <commit>
git revert--no-edit	Use the default commit message.			git revert --no-edit <commit>
git update-index	Register changes to the index.	None	--add, --remove, --skip-worktree, etc.	git update-index [options]
git update-index--add	Add files to the index.			git update-index --add <file>
git update-index--remove	Remove files from the index.			git update-index --remove <file>
git update-index--skip-worktree	Mark a file as "skip-worktree".			git update-index --skip-worktree <file>
git read-tree	Read a tree object into the index.	<tree>	--prefix=<prefix>, --index, --worktree	git read-tree [options] <tree>
git read-tree--prefix=<prefix>	Apply a prefix to files read from the tree.			git read-tree --prefix=<prefix> <tree>
git read-tree--index	Read the tree into the index only.			git read-tree --index <tree>
git read-tree--worktree	Apply the tree to the working directory.			git read-tree --worktree <tree>
git checkout-index	Check out files from the index.	None	--all, --force, --quiet	git checkout-index [options]
git checkout-index--all	Check out all files from the index.			git checkout-index --all
git checkout-index--force	Force the check-out of files.			git checkout-index --force <file>
git checkout-index--quiet	Suppress output.			git checkout-index --quiet

git commit-graph	Manage commit-graph files for performance improvements.	<subcommand>	--write, --verify, --expire, etc.	git commit-graph <subcommand> [options]
git commit-graph--write	Write the commit-graph file.			git commit-graph --write
git commit-graph--verify	Verify the commit-graph file.			git commit-graph --verify
git commit-graph--expire	Expire old commit-graph files.			git commit-graph --expire <time>
git commit-tree	Create a new tree object in the repository.	<tree>	-p <parent> (optional), -m <message> (commit message), etc.	git commit-tree [options] <tree>
git commit-tree-p <parent>	Specify the parent commit.			git commit-tree <tree> -p <parent>
git commit-tree-m <message>	Specify the commit message.			git commit-tree <tree> -m <message>
git hash-object	Compute the object ID and optionally create a blob from a file.	<file>	-w, --write, -t <type>, --stdin, --no-filename	git hash-object [options] <file>
git hash-object<file>	Compute the object ID for a file or create a blob from a file.			git hash-object <file>
git hash-object-w, --write	Write the object to the database.			git hash-object -w <file>
git hash-object-t <type>	Specify the type of the object (e.g., blob, tree, commit).			git hash-object -t <type> <file>

git hash-object--stdin	Read the object data from stdin.			git hash-object --stdin
git hash-object--no-filename	Do not show the filename in the output.			git hash-object --no-filename
git index-pack	Create an index for a pack file.	<packfile>	None	git index-pack [options] <packfile>
git merge-file	Merge changes between files.	<file1> <file2> <file3>	-p, --preserve	git merge-file [options] <file1> <file2> <file3>
git merge-file-p, --preserve	Preserve the contents of the output file.			git merge-file -p <file1> <file2> <file3>
git merge-index	Merge the index into the working directory.	None	None	git merge-index
git mktag	Create a tag object.	<tag>	None	git mktag [options] <tag>
git mktree	Create a new tree object from the index.	None	None	git mktree
git multi-pack-index	Manage multiple pack indexes.	<subcommand>	--write, --verify, --expire, etc.	git multi-pack-index <subcommand> [options]
git multi-pack-index--write	Write a new multi-pack index file.			git multi-pack-index --write
git multi-pack-index--verify	Verify the multi-pack index file.			git multi-pack-index --verify
git multi-pack-index--expire	Expire old multi-pack index files.			git multi-pack-index --expire <time>
git pack-objects	Create a pack file with the objects specified.	None	-c, --compression, -o <file>, --all, --no-reuse-deltas, etc.	git pack-objects [options]

git pack-objects-c, --compression	Specify compression level for the pack file.			git pack-objects -c <compression>
git pack-objects-o <file>	Output the pack file to the specified file.			git pack-objects -o <file>
git pack-objects--all	Include all objects.			git pack-objects --all
git pack-objects--no-reuse-deltas	Do not reuse deltas from other packs.			git pack-objects --no-reuse-deltas
git prune-packed	Remove unreachable objects from the packed object store.	None	None	git prune-packed
git replay	Replay commits from one branch to another.	<start> <end>	None	git replay <start> <end>
git symbolic-ref	Read or modify symbolic refs.	<ref>	-q (quiet)	git symbolic-ref <ref>
git symbolic-ref -q (quiet)	Shows the symbolic reference for a branch or a ref, and suppresses error messages if the ref is not found.			git symbolic-ref -q <ref>
git unpack-objects	Unpack objects from the given file.	<file>	None	git unpack-objects <file>
git update-ref	Update a ref to a specified value.	<ref> <value>	None	git update-ref <ref> <value>
git write-tree	Write the index as a tree object.	None	None	git write-tree
git cat-file	Provide content or type information for repository objects.	<type> <object>	None	git cat-file <type> <object>
git cherry	Find commits in one branch that are not in another.	<upstream> <branch>	None	git cherry <upstream> <branch>

git diff-files	Show changes between the working directory and index.	None	None	git diff-files
git diff-index	Show changes between the index and a tree.	<tree>	None	git diff-index <tree>
git diff-tree	Show changes between trees.	<tree> <tree>	None	git diff-tree <tree> <tree>
git for-each-ref	Iterate over each ref in the repository.	None	--format=<format>	git for-each-ref [<options>]
git for-each-ref--format=<format>	Format the output using the specified format.			git for-each-ref --format=<format>
git for-each-repo	Iterate over multiple repositories.	None	--format=<format>	git for-each-repo [<options>]
git for-each-repo--format=<format>	Format the output using the specified format for each repo.			git for-each-repo --format=<format>
git get-tar-commit-id	Get commit ID for a tarball.	<tarball>	None	git get-tar-commit-id <tarball>
git ls-files	Show information about files in the index.	None	--stage, --cached, --deleted	git ls-files [<options>]
git ls-files--stage	Show the stage number of each file in the index.			git ls-files --stage
git ls-files--cached	Show only files that are in the index (cached).			git ls-files --cached
git ls-files--deleted	Show deleted files.			git ls-files --deleted
git ls-remote	List references in a remote repository.	<repository>	None	git ls-remote <repository>

git ls-tree	List the contents of a tree object.	<tree>	-r (recursive), -t (list trees), -l (list files)	git ls-tree [<options>] <tree>
git ls-tree-r (recursive)	Recursively list all objects in a tree.			git ls-tree -r <tree>
git ls-tree-t (list trees)	List only tree objects.			git ls-tree -t <tree>
git ls-tree-l (list files)	List only file objects.			git ls-tree -l <tree>
git merge-base	Find the best common ancestor between two commits.	<commit1> <commit2>	None	git merge-base <commit1> <commit2>
git name-rev	Show the name of the most recent tag for a commit.	<commit>	None	git name-rev <commit>
git pack-redundant	Find redundant objects in a pack.	None	None	git pack-redundant
git rev-list	List commit objects in reverse chronological order.	<commit>	--max-count=<n>, --skip=<n>	git rev-list [<options>] <commit>
git rev-list--max-count=<n>	Limit the output to the first <n> commits.			git rev-list --max-count=<n>
git rev-list--skip=<n>	Skip the first <n> commits.			git rev-list --skip=<n>
git rev-parse	Parse and resolve git revision expressions.	<revision>	None	git rev-parse <revision>
git show-index	Show the contents of the index file.	None	None	git show-index
git show-ref	Show references in the repository.	None	-d (show detached HEAD)	git show-ref [<options>]
git show-ref-d (show detached HEAD)	Show the detached HEAD reference in the output.			git show-ref -d

git unpack-file	Unpack a file from the object store.	<file>	None	git unpack-file <file>
git var	Print a Git variable value.	<variable>	None	git var <variable>
git verify-pack	Verify the integrity of a pack file.	<pack>	None	git verify-pack <pack>
git fetch-pack	Fetch objects from a remote repository pack.	<repository>	None	git fetch-pack <repository>
git http-backend	Handle HTTP requests for Git repositories.	None	None	git http-backend
git send-pack	Send a pack of objects to a remote repository.	<repository>	None	git send-pack <repository>
git update-server-info	Update the server information files.	None	None	git update-server-info
git http-fetch	Fetch objects using HTTP.	<repository>	None	git http-fetch <repository>
git http-push	Push objects using HTTP.	<repository>	None	git http-push <repository>
git receive-pack	Receive objects from a remote repository.	None	None	git receive-pack
git shell	Run a Git shell.	None	None	git shell
git upload-archive	Create an archive of the repository.	<format> <path>	None	git upload-archive <format> <path>
git upload-pack	Upload objects to a remote repository pack.	None	None	git upload-pack
git check-attr	Check the attributes for files.	<attribute> <path>	None	git check-attr <attribute> <path>
git check-ignore	Check if files are ignored.	<file>	None	git check-ignore <file>
git check-mailmap	Check the mailmap configuration.	None	None	git check-mailmap

git check-ref-format	Check if a ref name is valid.	<ref>	None	git check-ref-format <ref>
git column	Format output into columns.	None	None	git column
git credential	Manage credentials for Git operations.	<command>	None	git credential <command>
git credential-cache	Manage credentials cache.	<command>	None	git credential-cache <command>
git credential-store	Manage credentials store.	<command>	None	git credential-store <command>
git fmt-merge-msg	Format merge commit messages.	None	None	git fmt-merge-msg
git hook	Manage Git hooks.	<hook>	None	git hook <hook>
git interpret-trailers	Interpret trailer lines in commit messages.	None	None	git interpret-trailers
git mailinfo	Read and process mail information.	<file>	None	git mailinfo <file>
git mailsplit	Split mail into individual messages.	<file>	None	git mailsplit <file>
git merge one file	Merges changes from one file into the current branch.	file	-	git merge --no-commit --no-ff file
git patch id	Displays or applies a patch identified by its ID.	id	-	git patch id <id>
git sh i18n	Shows internationalization (i18n) related information.	-	-	git sh i18n
git sh setup	Shows setup related information or performs setup actions.	-	-	git sh setup

git stripspace	Strips extra whitespace from lines of input.	-	-	git stripspace [<file>]
gitcore tutorial	Provides a tutorial on core Git concepts.	-	-	gitcore tutorial
gitcredentials	Manage credentials for Git.	-	-	git credentials [<options>]
gitcvs migration	Provides guidance on migrating from CVS to Git.	-	-	gitcvs migration
Git for CVS users	Tutorial or documentation for CVS users transitioning to Git.	-	-	git cvs tutorial
gitdiffcore	Core utilities for diff operations.	-	-	git diffcore
giteveryday	A guide to everyday Git usage.	-	-	giteveryday
gitfaq	Frequently asked questions about Git.	-	-	git faq
gitglossary	Provides a glossary of Git terms.	-	-	git glossary
A Git Glossary	Provides a glossary of Git terms (alternative command).	-	-	a git glossary
gitnamespaces	Provides information on Git namespaces.	-	-	git namespaces
gitremote helpers	Lists or manages remote helpers.	-	-	git remote-helpers [<options>]
gitsubmodules	Manages submodules in a Git repository.	add, status, update, etc.	-	git submodule [<command>] [<options>]
git submodule add	Adds a new submodule to the repository.			git submodule add <repository> [<path>]

git submodule status	Displays the current status of submodules.			git submodule status
git submodule update	Updates the submodules to match the commit specified in the main repository.			git submodule update [--init] [--recursive] [--remote]
gittutorial	Provides a tutorial on Git basics.	-	-	git tutorial
gittutorial 2	Provides an advanced tutorial on Git.	-	-	git tutorial 2
gitworkflows	Discusses various Git workflows.	-	-	git workflows
gitattributes	Manages attributes for files in the repository.	-	-	git attributes [<options>]
gitcli	Command-line interface for Git.	-	-	git cli
githooks	Manages Git hooks, which are scripts that run automatically on certain Git events.	-	-	git hooks
gitignore	Specifies files and directories to be ignored by Git.	-	-	git ignore [<options>]
gitmailmap	Maps email addresses to names.	-	-	git mailmap
gitmodules	Manages the configuration for submodules.	-	-	git modules
gitrepository layout	Provides the layout of the Git repository.	-	-	git repository layout

gitrevisions	Provides information on revisions in the Git repository.	-	-	git revisions
gitformat bundle	Formats a Git bundle.	-	-	git format bundle
gitformat chunk	Formats a Git chunk.	-	-	git format chunk
gitformat commit graph	Formats the commit graph for Git.	-	-	git format commit-graph
gitformat index	Formats the Git index.	-	-	git format index
gitformat pack	Formats Git pack files.	-	-	git format pack
gitformat signature	Formats Git signatures.	-	-	git format signature
gitprotocol capabilities	Shows capabilities of Git protocols.	-	-	git protocol capabilities
gitprotocol common	Provides information on common Git protocols.	-	-	git protocol common
gitprotocol http	Provides information on HTTP Git protocols.	-	-	git protocol http
Git HTTP based protocols	Details Git protocols based on HTTP.	-	-	git http-protocols
gitprotocol pack	Provides information on pack protocols in Git.	-	-	git protocol pack
gitprotocol v2	Provides information on Git protocol version 2.	-	-	git protocol v2
GIT_INDEX_FILE	Environment variable for specifying the index file.	path	-	GIT_INDEX_FILE=<path>
GIT_INDEX_VERSION	Environment variable specifying the index version.	version	-	GIT_INDEX_VERSION=<version>

GIT_OBJECT_DIRECTORY	Environment variable specifying the object directory.	path	-	GIT_OBJECT_DIRECTORY=<path>
GIT_ALTERNATE_OBJECT_DIRECTORIES	Environment variable for specifying alternate object directories.	path	-	GIT_ALTERNATE_OBJECT_DIRECTORIES=<path>
GIT_DIR	Environment variable for specifying the Git directory.	path	-	GIT_DIR=<path>
GIT_WORK_TREE	Environment variable for specifying the working tree directory.	path	-	GIT_WORK_TREE=<path>
GIT_NAMESPACE	Environment variable for specifying the Git namespace.	namespace	-	GIT_NAMESPACE=<namespace>
GIT_CEILING_DIRECTORIES	Environment variable for specifying ceiling directories.	path	-	GIT_CEILING_DIRECTORIES=<path>
GIT_DISCOVERY_ACROSS_FILESYSTEM	Environment variable to control cross-filesystem discovery.	true or false	-	GIT_DISCOVERY_ACROSS_FILESYSTEM=<value>
GIT_COMMON_DIR	Environment variable for specifying the common directory.	path	-	GIT_COMMON_DIR=<path>
GIT_DEFAULT_HASH	Specifies the default hash algorithm used by Git.		SHA1, SHA256, etc.	GIT_DEFAULT_HASH=<algorithm>
GIT_DEFAULT_REF_FORMAT	Specifies the default format for ref names in Git.		short, full, etc.	GIT_DEFAULT_REF_FORMAT=<format>
GIT_AUTHOR_NAME	Name of the author of the commit.		String	GIT_AUTHOR_NAME=<name>
GIT_AUTHOR_EMAIL	Email address of the author of the commit.		Email address	GIT_AUTHOR_EMAIL=<email>

GIT_AUTHOR_DATE	Date of the author’s commit.		Date in ISO 8601 format (e.g., 2024-07-20T12:00:00Z)	GIT_AUTHOR_DATE=<date>
GIT_COMMITTER_NAME	Name of the committer of the commit.		String	GIT_COMMITTER_NAME=<name>
GIT_COMMITTER_EMAIL	Email address of the committer of the commit.		Email address	GIT_COMMITTER_EMAIL=<email>
GIT_COMMITTER_DATE	Date of the committer’s commit.		Date in ISO 8601 format (e.g., 2024-07-20T12:00:00Z)	GIT_COMMITTER_DATE=<date>
GIT_DIFF_OPTS	Options for customizing git diff output.		Options for diff command (e.g., --color, --stat, etc.)	GIT_DIFF_OPTS=<options>
GIT_EXTERNAL_DIFF	Command to use for external diffing.		Command line arguments for external diff tool	GIT_EXTERNAL_DIFF=<command>
GIT_DIFF_PATH_COUNTER	Path counter in the diff output.		Integer	GIT_DIFF_PATH_COUNTER=<number>
GIT_DIFF_PATH_TOTAL	Total number of paths in the diff output.		Integer	GIT_DIFF_PATH_TOTAL=<number>
GIT_MERGE_VERBOSITY	Controls the verbosity of merge output.		Level of verbosity (e.g., 0 for minimal, 1 for default, 2 for verbose)	GIT_MERGE_VERBOSITY=<level>
GIT_PAGER	Pager program to use for viewing output.		Pager command (e.g., less, more, etc.)	GIT_PAGER=<pager>
GIT_PROGRESS_DELAY	Delay in seconds before displaying progress information.		Integer	GIT_PROGRESS_DELAY=<seconds>

GIT_EDITOR	Editor to use for commit messages and other Git interactions.		Editor command (e.g., vim, nano, etc.)	GIT_EDITOR=<editor>
GIT_SEQUENCE_EDITOR	Editor to use for interactive rebase sequences.		Editor command (e.g., vim, nano, etc.)	GIT_SEQUENCE_EDITOR=<editor>
GIT_SSH	SSH command to use for Git operations over SSH.		SSH command (e.g., ssh, ssh -i keyfile, etc.)	GIT_SSH=<ssh-command>
GIT_SSH_COMMAND	Command to use for SSH operations.		SSH command with arguments (e.g., ssh -o StrictHostKeyChecking=no)	GIT_SSH_COMMAND=<command>
GIT_SSH_VARIANT	Variant of SSH to use (e.g., ssh, plink).		SSH variant	GIT_SSH_VARIANT=<variant>
GIT_SSL_NO_VERIFY	Disables SSL verification for Git commands.		true or false	`GIT_SSL_NO_VERIFY=<true
GIT_ATTR_SOURCE	Specifies the source of Git attributes.		Path to the attributes file or default for default attributes	`GIT_ATTR_SOURCE=<path
GIT_ASKPASS	Program to use for prompts asking for credentials.		Program command (e.g., ssh-askpass, git-askpass)	GIT_ASKPASS=<program>
GIT_TERMINAL_PROMPT	Controls whether Git prompts for credentials in the terminal.		true or false	`GIT_TERMINAL_PROMPT=<true
GIT_CONFIG_GLOBAL	Path to the global Git configuration file.		File path	GIT_CONFIG_GLOBAL=<path>

GIT_CONFIG_SYSTEM	Path to the system-wide Git configuration file.		File path	GIT_CONFIG_SYSTEM=<path>
GIT_CONFIG_NOSYSTEM	Disables the system-wide Git configuration file.		true or false	`GIT_CONFIG_NOSYSTEM=<true
GIT_FLUSH	Controls whether to flush the output buffer after each command.		true or false	`GIT_FLUSH=<true
GIT_TRACE	Enables tracing of Git operations.		Trace level (e.g., 1, 2)	GIT_TRACE=<level>
GIT_TRACE_FSMONITOR	Enables tracing of filesystem monitoring.		true or false	`GIT_TRACE_FSMONITOR=<true
GIT_TRACE_PACK_ACCESS	Enables tracing of packfile access.		true or false	`GIT_TRACE_PACK_ACCESS=<true
GIT_TRACE_PACKET	Enables tracing of Git packet communication.		true or false	`GIT_TRACE_PACKET=<true
GIT_TRACE_PACKFILE	Enables tracing of packfile operations.		true or false	`GIT_TRACE_PACKFILE=<true
GIT_TRACE_PERFORMANCE	Enables tracing of performance metrics.		true or false	`GIT_TRACE_PERFORMANCE=<true
GIT_TRACE_REFS	Enables tracing of reference updates.		true or false	`GIT_TRACE_REFS=<true
GIT_TRACE_SETUP	Enables tracing of Git setup operations.		true or false	`GIT_TRACE_SETUP=<true
GIT_TRACE_SHALLOW	Enables tracing of shallow clone operations.		true or false	`GIT_TRACE_SHALLOW=<true
GIT_TRACE_CURL	Enables tracing of curl operations used by Git.		true or false	`GIT_TRACE_CURL=<true

GIT_TRACE_CURL_NO_DATA	Enables tracing of curl operations without data.		true or false	`GIT_TRACE_CURL_NO_DATA=<true
GIT_TRACE2	Enables tracing of Git operations with the trace2 format.		true or false	`GIT_TRACE2=<true
GIT_TRACE2_EVENT	Enables tracing of trace2 events.		true or false	`GIT_TRACE2_EVENT=<true
GIT_TRACE2_PERF	Enables tracing of trace2 performance metrics.		true or false	`GIT_TRACE2_PERF=<true
GIT_TRACE_REDACT	Controls the redaction of sensitive data in trace logs.		true or false	`GIT_TRACE_REDACT=<true
GIT_NO_REPLACE_OBJECTS	Disables replacement objects for Git commands.		true or false	`GIT_NO_REPLACE_OBJECTS=<true
GIT_LITERAL_PATHSPECS	Enables literal path specifications.		true or false	`GIT_LITERAL_PATHSPECS=<true
GIT_GLOB_PATHSPECS	Enables glob path specifications.		true or false	`GIT_GLOB_PATHSPECS=<true
GIT_NOGLOB_PATHSPECS	Disables glob path specifications.		true or false	`GIT_NOGLOB_PATHSPECS=<true
GIT_ICASE_PATHSPECS	Enables case-insensitive path specifications.		true or false	`GIT_ICASE_PATHSPECS=<true
GIT_NO_LAZY_FETCH	Disables lazy fetches in Git operations, which may affect performance but ensure more consistent results.		1 to disable lazy fetches.	GIT_NO_LAZY_FETCH=1

GIT_REFLOG_ACTION	Used to specify an action to include in the reflog.		action (e.g., commit, checkout).	GIT_REFLOG_ACTION=action
GIT_REF_PARANOIA	Sets the paranoia level for Git references.		level (e.g., 0, 1, 2).	GIT_REF_PARANOIA=level
GIT_COMMIT_GRAPH_PARANOIA	Controls the paranoia level for commit graph.		level (e.g., 0, 1, 2).	GIT_COMMIT_GRAPH_PARANOIA=level
GIT_ALLOW_PROTOCOL	Specifies which protocols Git is allowed to use.		protocol (e.g., http, https, ssh).	GIT_ALLOW_PROTOCOL=protocol
GIT_PROTOCOL_FROM_USER	Specifies the protocol to use if the user has not set it.		protocol (e.g., http, https, ssh).	GIT_PROTOCOL_FROM_USER=protocol
GIT_PROTOCOL	Defines the protocol to use for Git operations.		protocol (e.g., http, https, ssh).	GIT_PROTOCOL=protocol
GIT_OPTIONAL_LOCKS	Determines if optional file locking is used in Git operations.		1 to enable, 0 to disable.	GIT_OPTIONAL_LOCKS=1 or GIT_OPTIONAL_LOCKS=0
GIT_REDIRECT_STDIN	Redirects standard input to a file or pipe.		path or - (to ignore).	GIT_REDIRECT_STDIN=path
GIT_REDIRECT_STDOUT	Redirects standard output to a file or pipe.		path or - (to ignore).	GIT_REDIRECT_STDOUT=path
GIT_REDIRECT_STDERR	Redirects standard error to a file or pipe.		path or - (to ignore).	GIT_REDIRECT_STDERR=path
GIT_PRINT_SHA1_ELLIPSIS (deprecated)	Controls the printing of shortened SHA-1 hashes in Git commands.		Not applicable (deprecated).	GIT_PRINT_SHA1_ELLIPSIS=value