

Sidebar

Programming · VS Code Git and Source Control Cheat Sheet

The `git.*` and `scm.*` settings and the everyday source control actions in VS Code. Exact ID, what it does, and its default. Sheet 8 in the VS Code series. This covers VS Code's own settings, not Git commands (see the Git & GitHub courses for those).

EVERYDAY SOURCE CONTROL

Ctrl+Shift+G Open the Source Control view

Ctrl+Enter Commit, when the cursor is in the commit message box

+ / - on a file Stage / unstage that file; the + on the Changes heading stages everything

Git: Initialize Repository Command Palette: turn the open folder into a Git repository

Git: Clone Command Palette: copy a remote repository to your machine

Git: Create Branch / Git: Checkout to... Make a new branch / switch branch (also: click the branch name in the status bar)

Git: Pull / Push / Fetch / Sync Commands for talking to the remote; *Sync* pulls then pushes

Git: Stash Put uncommitted changes aside (Git: Pop Stash brings them back)

FETCHING, PULLING & SYNCING

git.autofetch **false** (default), **true**, **all**: fetch from the remote in the background

git.autofetchPeriod Seconds between background fetches (default 180)

git.autorefresh Refresh the Source Control view when files change (default **true**)

git.confirmSync Ask before syncing (default **true**)

git.fetchOnPull Fetch everything when you pull (default **false**)

git.pruneOnFetch Remove remote branches that no longer exist (default **false**)

git.rebaseWhenSync Rebase instead of merge when you sync (default **false**)

git.autoStash Stash and restore changes automatically around a pull (default **false**)

git.postCommitCommand After committing: **none** (default), **push**, **sync**

COMMITTING

`git.enableSmartCommit` Commit all changes when nothing is staged (default `false`)

`git.smartCommitChanges` `all` (default) or `tracked`: which changes a smart commit includes

`git.suggestSmartCommit` Offer smart commit when nothing is staged (default `true`)

`git.promptToSaveFilesBeforeCommit` `always` (default), `staged`, `never`: save unsaved files first

`git.inputValidation` Warn when the commit message is too long (default `false`)

`git.inputValidationSubjectLength` First-line length that triggers the warning (default 50)

`git.inputValidationLength` Overall line length that triggers the warning (default 72)

`git.verboseCommit` Show the diff in the commit message editor (default `false`)

`git.enableCommitSigning` Sign commits with GPG or SSH (default `false`)

`git.requireGitUserConfig` Insist that `user.name` and `user.email` are set (default `true`)

BRANCHES & CLONING

`git.defaultBranchName` Name for the first branch of a new repository (default `main`)

`git.branchPrefix` Text added in front of new branch names, e.g. `feature/` (default empty)

`git.branchSortOrder` `committerdate` (default) or `alphabetically`

`git.branchValidationRegex` Pattern that new branch names must match (default none)

`git.checkoutType` What the checkout list shows (default `local`, `remote`, `tags`)

`git.openAfterClone` `prompt` (default), `always`, `alwaysNewWindow`, `whenNoFolderOpen`

`git.defaultCloneDirectory` Folder cloned repositories go into (default none)

`git.showReferenceDetails` Show last commit details in the branch picker (default `true`)

PUSHING

`git.allowForcePush` Enable force-push commands (default `false`)

`git.useForcePushWithLease` `true` (default): force-push only if nobody else has pushed since you last fetched

`git.confirmForcePush` Ask before force-pushing (default `true`)

`git.showPushSuccessNotification` Show a message after a successful push (default `false`)

WHAT THE GIT EXTENSION SHOWS

`git.untrackedChanges` `mixed` (default), `separate`, `hidden`: where new untracked files appear

`git.openDiffOnClick` Clicking a changed file opens its diff (default `true`)

`git.decorations.enabled` Colours and badges on files in the Explorer (default `true`)

`git.showInlineOpenFileAction` *Open File* button on each change (default `true`)

`git.detectSubmodules` Detect submodules (default `true`)

`git.repositoryScanMaxDepth` How deep VS Code looks for repositories in subfolders (default 1)

`git.ignoredRepositories` / `git.scanRepositories` Repositories to ignore / extra folders to scan (default none)

`git.ignoreLimitWarning` Silence the warning about too many changes (default `false`)

`git.enabled` / `git.path` Turn the extension off, or point it at a specific `git` program (defaults: on, auto-detected)

SOURCE CONTROL VIEW (SCM.*)

`scm.defaultViewMode` `list` (default) or `tree`

`scm.defaultViewSortKey` `path` (default), `name`, `status`

`scm.compactFolders` Compact single-child folder chains in tree mode (default `true`)

`scm.autoReveal` Select the active file in the view (default `true`)

`scm.countBadge` Badge on the activity bar: `all` (default), `focused`, `off`

`scm.diffDecorations` Change marks in the editor: `all` (default), `gutter`, `overview`, `minimap`, `none`

`scm.alwaysShowActions` Always show inline action buttons (default `false`)

`scm.alwaysShowRepositories` Always list the repositories pane (default `false`)

`scm.graph.pageSize` Commits loaded at a time in the graph (default 50)

TIMELINE, BLAME & DIFFS

`git.timeline.showAuthor` Show the author in the Timeline view (default `true`)

`git.timeline.date` `committed` (default) or `authored`

`git.blame.editorDecoration.enabled` Show who last changed each line, inline (default `false`)

`git.blame.statusBarItem.enabled` Blame for the current line in the status bar (default `true`)

`diffEditor.renderSideBySide` `true`: side-by-side diff; `false`: inline

`diffEditor.ignoreTrimWhitespace` Ignore whitespace-only changes at line ends in diffs

MERGE CONFLICTS

Accept Current / Incoming / Both

The links shown above each conflict block in the editor

Compare Changes

Link above a conflict: opens the two versions side by side

`git.mergeEditor`

`true`: open conflicts in the three-way merge editor (default)
`false` in this version)

`mergeEditor.diffAlgorithm`

`advanced` (default) or `legacy` (plus external/wasm variants)

Resolve in Merge Editor

Button on a conflicted file, available when `git.mergeEditor` is on

A REASONABLE STARTER SET (MY SUGGESTIONS, NOT DEFAULTS)

`"git.autofetch": true`

Always know whether the remote moved

`"git.confirmSync": false`

Skip the sync prompt once you trust the habit

`"git.pruneOnFetch": true`

Drop remote branches that were deleted

`"git.inputValidation": true`

Nudges you towards short subject lines

`"git.blame.editorDecoration.enabled": true`

Inline blame for who changed what

`"git.mergeEditor": true`

A clearer way to resolve conflicts

`"git.enableSmartCommit": false`

Keep commits deliberate (this is the default)

WHY DID SYNC PROMPT ME, AND WHAT DID IT DO?

Sync Changes runs a pull and then a push. With `git.confirmSync` on (the default) VS Code asks first, because a pull can change your files. If you prefer to rebase rather than merge when pulling, turn on `git.rebaseWhenSync`. If you would rather see what is coming before pulling, use *Git: Fetch* or turn on `git.autofetch` and check the Source Control view.

FORCE PUSH, CAREFULLY

Force-pushing is off by default (`git.allowForcePush`). If you turn it on, keep `git.useForcePushWithLease` on: the push then fails if someone else has pushed new commits since your last fetch, instead of overwriting their work.