# fetch

`fetch.recurseSubmodules`
This option controls whether `git` `fetch` (and the underlying fetch in `git` `pull`) will recursively fetch into populated submodules. This option can be set either to a boolean value or to `on-demand`. Setting it to a boolean changes the behavior of fetch and pull to recurse unconditionally into submodules when set to true or to not recurse at all when set to false. When set to `on-demand`, fetch and pull will only recurse into a populated submodule when its superproject retrieves a commit that updates the submodule's reference. Defaults to `on-demand`, or to the value of `submodule.recurse` if set.

`fetch.fsckObjects`
If it is set to true, git-fetch-pack will check all fetched objects. See `transfer.fsckObjects` for what's checked. Defaults to `false`. If not set, the value of `transfer.fsckObjects` is used instead.

`fetch.fsck.<msg-id>`
Acts like `fsck.<msg-id>`, but is used by linkgit:git-fetch-pack[1] instead of linkgit:git-fsck[1]. See the `fsck.<msg-id>` documentation for details.

`fetch.fsck.skipList`
Acts like `fsck.skipList`, but is used by linkgit:git-fetch-pack[1] instead of linkgit:git-fsck[1]. See the `fsck.skipList` documentation for details.

`fetch.unpackLimit`
If the number of objects fetched over the Git native transfer is below this limit, then the objects will be unpacked into loose object files. However if the number of received objects equals or exceeds this limit then the received pack will be stored as a pack, after adding any missing delta bases. Storing the pack from a push can make the push operation complete faster, especially on slow filesystems. If not set, the value of `transfer.unpackLimit` is used instead.

`fetch.prune`
If true, fetch will automatically behave as if the `--prune` option was given on the command line. See also `remote.<name>.prune` and the PRUNING section of linkgit:git-fetch[1].

`fetch.pruneTags`
If true, fetch will automatically behave as if the `refs/tags/` `:refs/tags/` refspec was provided when pruning, if not set already. This allows for setting both this option and `fetch.prune` to maintain a 1=1 mapping to upstream refs. See also `remote.<name>.pruneTags` and the PRUNING section of linkgit:git-fetch[1].

`fetch.all`
If true, fetch will attempt to update all available remotes. This behavior can be overridden by passing `--no-all` or by explicitly specifying one or more remote(s) to fetch from. Defaults to `false`

.

`fetch.output`
Control how ref update status is printed. Valid values are `full` and `compact`. Default value is `full`. See the OUTPUT section in linkgit:git-fetch[1] for details.

`fetch.negotiationAlgorithm`
Control how information about the commits in the local repository is sent when negotiating the contents of the packfile to be sent by the server. Set to `consecutive` to use an algorithm that walks over consecutive commits checking each one. Set to `skipping` to use an algorithm that skips commits in an effort to converge faster, but may result in a larger-than-necessary packfile; or set to `noop` to not send any information at all, which will almost certainly result in a larger-than-necessary packfile, but will skip the negotiation step. Set to `default` to override settings made previously and use the default behaviour. The default is normally `consecutive`, but if `feature.experimental` is `true`, then the default is `skipping`. Unknown values will cause `git` `fetch` to error out.

See also the `--negotiate-only` and `--negotiation-restrict` options to linkgit:git-fetch[1].

`fetch.showForcedUpdates`
Set to `false` to enable `--no-show-forced-updates` in linkgit:git-fetch[1] and linkgit:git-pull[1] commands. Defaults to `true`.

`fetch.parallel`
Specifies the maximal number of fetch operations to be run in parallel at a time (submodules, or remotes when the `--multiple` option of linkgit:git-fetch[1] is in effect).

A value of 0 will give some reasonable default. If unset, it defaults to 1.

For submodules, this setting can be overridden using the `submodule.fetchJobs` config setting.

`fetch.writeCommitGraph`
Set to true to write a commit-graph after every `git` `fetch` command that downloads a pack-file from a remote. Using the `--split` option, most executions will create a very small commit-graph file on top of the existing commit-graph file(s). Occasionally, these files will merge and the write may take longer. Having an updated commit-graph file helps performance of many Git commands, including `git` `merge-base`, `git` `push` `-f`, and `git` `log` `--graph`. Defaults to `false`.

`fetch.bundleURI`
This value stores a URI for downloading Git object data from a bundle URI before performing an incremental fetch from the origin Git server. This is similar to how the `--bundle-uri` option behaves in linkgit:git-clone[1]. `git` `clone` `--bundle-uri` will set the `fetch.bundleURI` value if the supplied bundle URI contains a bundle list that is organized for incremental fetches.

If you modify this value and your repository has a `fetch.bundleCreationToken` value, then remove that `fetch.bundleCreationToken` value before fetching from the new bundle URI.

`fetch.bundleCreationToken`
When using `fetch.bundleURI` to fetch incrementally from a bundle list that uses the “creationToken” heuristic, this config value stores the maximum `creationToken` value of the downloaded bundles. This value is used to prevent downloading bundles in the future if the advertised `creationToken` is not strictly larger than this value.

The creation token values are chosen by the provider serving the specific bundle URI. If you modify the URI at `fetch.bundleURI`, then be sure to remove the value for the `fetch.bundleCreationToken` value before fetching.

`fetch.followRemoteHEAD`
When fetching using a default refspec, this setting determines how to handle differences between a fetched remote’s `HEAD` and the local `remotes/<name>/HEAD` symbolic-ref. Its value is one of

`create`;; Create `remotes/<name>/HEAD` if a ref exists on the remote, but not locally. An existing symbolic-ref will not be touched. This is the default value. `warn`;; Display a warning if the remote advertises a different `HEAD` than what is set locally. Behaves like "create" if the local symbolic-ref doesn’t exist. `always`;; Silently update `remotes/<name>/HEAD` whenever the remote advertises a new value. `never`;; Never create or modify the `remotes/<name>/HEAD` symbolic-ref.

---

□□□ #2
□ □□□ □□□ 2026-10-31 20:06:45 UTC
□ □□□ □□□ 2026-10-31 20:06:47 UTC