

NAME

git-remote - Manage set of tracked repositories

SYNOPSIS

```
git remote [-v | --verbose]
git remote add[-t <branch h>] [-m <master>] [-f] [--[no-]tags] [--mirror=<fetch|push>] <name> <url>
git remote rename <old> <new>
git remote remove <name>
git remote set-head <name> (-a | --auto | -d | --delete | <branch>)
git remote set-branches [--add] <name> <branch>...
git remote set-url[--push] <name> <newurl> [<oldurl>]
git remote set-url --add[--push] <name> <newurl>
git remote set-url --delete[--push] <name> <url>
git remote [-v | --verbose] show [-n] <name>...
git remote prune[-n | --dry-run] <name>...
git remote [-v | --verbose] update [-p | --prune] [(<group> | <remote>)...]
```

DESCRIPTION

Manage the set of repositories (remotes) whose branches you track.

OPTIONS

-v, --verbose

Be a little more verbose and show remote url after name. NOTE: This must be placed between remote and subcommand.

COMMANDS

With no arguments, shows a list of existing remotes. Several subcommands are available to perform operations on the remotes.

add

Adds a remote named <name> for the repository at <url>. The command `git fetch <name>` can then be used to create and update remote-tracking branches <name>/<branch>.

With -f option, `git fetch <name>` is run immediately after the remote information is set up.

With --tags option, `git fetch <name>` imports every tag from the remote repository.

With --no-tags option, `git fetch <name>` does not import tags from the remote repository.

With -t <branch> option, instead of the default glob refspec for the remote to track all branches under the refs/remotes/<name>/ namespace, a refspec to track only <branch> is created. You can give more than one -t <branch> to track multiple branches without grabbing all branches.

With -m <master> option, a symbolic-ref refs/remotes/<name>/HEAD is set up to point at remote's <master> branch. See also the set-head command.

When a fetch mirror is created with --mirror=fetch, the refs will not be stored in the refs/remotes/ namespace, but rather everything in refs/ on the remote will be directly mirrored into refs/ in the local repository. This option only makes sense in bare repositories, because a fetch would overwrite any local commits.

When a push mirror is created with --mirror=push, then `git push` will always behave as if --mirror was passed.

rename

Rename the remote named <old> to <new>. All remote-tracking branches and configuration settings for the remote are updated.

In case <old> and <new> are the same, and <old> is a file under \$GIT_DIR/remotes or \$GIT_DIR/branches, the remote is converted to the configuration file format.

remove, rm

Remove the remote named `<name>`. All remote-tracking branches and configuration settings for the remote are removed.

set-head

Sets or deletes the default branch (i.e. the target of the symbolic-ref `refs/remotes/<name>/HEAD`) for the named remote. Having a default branch for a remote is not required, but allows the name of the remote to be specified in lieu of a specific branch. For example, if the default branch for origin is set to master, then origin may be specified wherever you would normally specify `origin/master`.

With `-d` or `--delete`, the symbolic ref `refs/remotes/<name>/HEAD` is deleted.

With `-a` or `--auto`, the remote is queried to determine its HEAD, then the symbolic-ref `refs/remotes/<name>/HEAD` is set to the same branch. e.g., if the remote HEAD is pointed at next, `git remote set-head origin -a` will set the symbolic-ref `refs/remotes/origin/HEAD` to `refs/remotes/origin/next`. This will only work if `refs/remotes/origin/next` already exists; if not it must be fetched first.

Use `<branch>` to set the symbolic-ref `refs/remotes/<name>/HEAD` explicitly. e.g., `git remote set-head origin master` will set the symbolic-ref `refs/remotes/origin/HEAD` to `refs/remotes/origin/master`. This will only work if `refs/remotes/origin/master` already exists; if not it must be fetched first.

set-branches

Changes the list of branches tracked by the named remote. This can be used to track a subset of the available remote branches after the initial setup for a remote.

The named branches will be interpreted as if specified with the `-t` option on the `git remote add` command line.

With `--add`, instead of replacing the list of currently tracked branches, adds to that list.

set-url

Changes URL remote points to. Sets first URL remote points to matching regex `<oldurl>` (first URL if no `<oldurl>` is given) to `<newurl>`. If `<oldurl>` doesn't match any URL, error occurs and nothing is changed.

With `--push`, push URLs are manipulated instead of fetch URLs.

With `--add`, instead of changing some URL, new URL is added.

With `--delete`, instead of changing some URL, all URLs matching regex `<url>` are deleted. Trying to delete all non-push URLs is an error.

show

Gives some information about the remote `<name>`.

With `-n` option, the remote heads are not queried first with `git ls-remote <name>`; cached information is used instead.

prune

Deletes all stale remote-tracking branches under `<name>`. These stale branches have already been removed from the remote repository referenced by `<name>`, but are still locally available in `remotes/<name>`.

With `--dry-run` option, report what branches will be pruned, but do not actually prune them.

update

Fetch updates for a named set of remotes in the repository as defined by `remotes.<group>`. If a named group is not specified on the command line, the configuration parameter `remotes.default` will be used; if `remotes.default` is not defined, all remotes which do not have the configuration parameter `remote.<name>.skipDefaultUpdate` set to true will be updated.

(See [git-config\(1\)](#)).

With `--prune` option, prune all the remotes that are updated.

DISCUSSION

The remote configuration is achieved using the `remote.origin.url` and `remote.origin.fetch` configuration variables. (See [git-config\(1\)](#)).

EXAMPLES

- Add a new remote, fetch, and check out a branch from it

```
$ git remote
origin
$ git branch -r
origin/HEAD -> origin/master
origin/master
$ git remote add staging git://git.kernel.org/.../gregkh/staging.git
$ git remote
origin
staging
$ git fetch staging
...
From git://git.kernel.org/pub/scm/linux/kernel/git/gregkh/staging
* [new branch] master -> staging/master
* [new branch] staging-linus -> staging/staging-linus
* [new branch] staging-next -> staging/staging-next
$ git branch -r
origin/HEAD -> origin/master
origin/master
staging/master
staging/staging-linus
staging/staging-next
$ git checkout -b staging staging/master
...
```

- Imitate *git clone* but track only selected branches

```
$ mkdir project.git
$ cd project.git
$ git init
$ git remote add -f -t master -m master origin git://example.com/git.git/
$ git merge origin
```

SEE ALSO

[git-fetch\(1\)](#) [git-branch\(1\)](#) [git-config\(1\)](#)

GIT

Part of the [git\(1\)](#) suite