Pushing commits to a remote repository

In this article

Use `git push` to push commits made on your local branch to a remote repository.

About `git push`

The `git push` command takes two arguments:

- A remote name, for example, `origin`
- A branch name, for example, `main`

For example:

```
git push REMOTE-NAME BRANCH-NAME
```

As an example, you usually run `git push origin main` to push your local changes to your online repository.

Renaming branches

To rename a branch, you'd use the same `git push` command, but you would add one more argument: the name of the new branch. For example:

```
git push REMOTE-NAME LOCAL-BRANCH-NAME:REMOTE-BRANCH-NAME
```

This pushes the `LOCAL-BRANCH-NAME` to your `REMOTE-NAME`, but it is renamed to `REMOTE-BRANCH-NAME`.

Dealing with "non-fast-forward" errors

If your local copy of a repository is out of sync with, or "behind," the upstream repository you're pushing to, you'll get a message saying `non-fast-forward updates were rejected`. This means that you must retrieve, or "fetch," the upstream changes, before you are able to push your local changes.

For more information on this error, see "Dealing with non-fast-forward errors."

Pushing tags

By default, and without additional parameters, `git push` sends all matching branches that have the same names as remote branches.

To push a single tag, you can issue the same command as pushing a branch:

```
git push REMOTE-NAME TAG-NAME
```

To push all your tags, you can type the command:

```
git push REMOTE-NAME --tags
```

Deleting a remote branch or tag

The syntax to delete a branch is a bit arcane at first glance:

```
git push REMOTE-NAME :BRANCH-NAME
```

Note that there is a space before the colon. The command resembles the same steps you'd take to rename a branch. However, here, you're telling Git to push *nothing* into `BRANCH-NAME` on `REMOTE-NAME`. Because of this, `git push` deletes the branch on the remote repository.

Remotes and forks

You might already know that you can "fork" repositories on GitHub.

When you clone a repository you own, you provide it with a remote URL that tells Git where to fetch and push updates. If you want to collaborate with the original repository, you'd add a new remote URL, typically called `upstream`, to your local Git clone:

```
git remote add upstream THEIR_REMOTE_URL
```

Now, you can fetch updates and branches from *their* fork:

```
git fetch upstream
# Grab the upstream remote's branches
> remote: Counting objects: 75, done.
> remote: Compressing objects: 100% (53/53), done.
> remote: Total 62 (delta 27), reused 44 (delta 9)
> Unpacking objects: 100% (62/62), done.
> From https://HOSTNAME/OCTOCAT/REPO
>  * [new branch]      main     -> upstream/main
```

When you're done making local changes, you can push your local branch to GitHub and initiate a pull request.

For more information on working with forks, see "Syncing a fork".

Further reading

- The "Remotes" chapter from the "Pro Git" book
- `git remote` main page
- "Git cheatsheet"
- "Git workflows"
- "Git Handbook"

Legal

© 2023 GitHub, Inc. Terms Privacy Status Pricing Expert services Blog