Managing remote repositories

Learn to work with your local repositories on your computer and remote repositories hosted on GitHub.

Mac	Windows	**Linux**

In this article

Adding a remote repository

To add a new remote, use the `git remote add` command on the terminal, in the directory your repository is stored at.

The `git remote add` command takes two arguments:

- A remote name, for example, `origin`

- A remote URL, for example, `https://github.com/user/repo.git`

For example:

```
$ git remote add origin https://github.com/user/repo.git
# Set a new remote

$ git remote -v
# Verify new remote
> origin  https://github.com/user/repo.git (fetch)
```

```
> origin  https://github.com/user/repo.git (push)
```

For more information on which URL to use, see "About remote repositories."

Troubleshooting: Remote origin already exists

This error means you've tried to add a remote with a name that already exists in your local repository.

```
$ git remote add origin https://github.com/octocat/Spoon-Knife.git
> fatal: remote origin already exists.
```

To fix this, you can:

- Use a different name for the new remote.
- Rename the existing remote repository before you add the new remote. For more information, see "Renaming a remote repository" below.

- Delete the existing remote repository before you add the new remote. For more information, see "Removing a remote repository" below.

Changing a remote repository's URL

The `git remote set-url` command changes an existing remote repository URL.

> **Tip:** For information on the difference between HTTPS and SSH URLs, see "About remote repositories."

The `git remote set-url` command takes two arguments:

- An existing remote name. For example, `origin` or `upstream` are two common choices.
- A new URL for the remote. For example:

  - If you're updating to use HTTPS, your URL might look like:

    ```
    https://github.com/USERNAME/REPOSITORY.git
    ```

  - If you're updating to use SSH, your URL might look like:

```
git@github.com:USERNAME/REPOSITORY.git
```

☐ GitHub Docs   ☰

1   Open Terminal.

2   Change the current working directory to your local project.

3   List your existing remotes in order to get the name of the remote you want to change.

```
$ git remote -v
> origin  git@github.com:USERNAME/REPOSITORY.git (fetch)
> origin  git@github.com:USERNAME/REPOSITORY.git (push)
```

4   Change your remote's URL from SSH to HTTPS with the `git remote set-url` command.

```
$ git remote set-url origin https://github.com/USERNAME/REPOSITORY.git
```

5   Verify that the remote URL has changed.

```
$ git remote -v
# Verify new remote URL
> origin  https://github.com/USERNAME/REPOSITORY.git (fetch)
> origin  https://github.com/USERNAME/REPOSITORY.git (push)
```

The next time you `git fetch`, `git pull`, or `git push` to the remote repository, you'll be asked for your GitHub username and password. When Git prompts you for your password, enter your personal access token (PAT) instead. Password-based authentication for Git has been removed, and using a PAT is more secure. For more information, see "Creating a personal access token."

You can use a credential helper so Git will remember your GitHub username and personal access token every time it talks to GitHub.

Switching remote URLs from HTTPS to SSH

1   Open Terminal.

2   Change the current working directory to your local project.

3   List your existing remotes in order to get the name of the remote you want to change.

```
$ git remote -v
> origin  https://github.com/USERNAME/REPOSITORY.git (fetch)
> origin  https://github.com/USERNAME/REPOSITORY.git (push)
```

4   Change your remote's URL from HTTPS to SSH with the `git remote set-url` command.

```
$ git remote set-url origin git@github.com:USERNAME/REPOSITORY.git
```

5   Verify that the remote URL has changed.

```
$ git remote -v
# Verify new remote URL
> origin  git@github.com:USERNAME/REPOSITORY.git (fetch)
> origin  git@github.com:USERNAME/REPOSITORY.git (push)
```

Troubleshooting: No such remote '[name]'

This error means that the remote you tried to change doesn't exist:

```
$ git remote set-url sofake https://github.com/octocat/Spoon-Knife
> fatal: No such remote 'sofake'
```

Check that you've correctly typed the remote name.

Renaming a remote repository

Use the `git remote rename` command to rename an existing remote.

The `git remote rename` command takes two arguments:

- An existing remote name, for example, `origin`
- A new name for the remote, for example, `destination`

Example

These examples assume you're [cloning using HTTPS](), which is recommended.

```
$ git remote -v
# View existing remotes
> origin  https://github.com/OWNER/REPOSITORY.git (fetch)
> origin  https://github.com/OWNER/REPOSITORY.git (push)

$ git remote rename origin destination
# Change remote name from 'origin' to 'destination'

$ git remote -v
# Verify remote's new name
> destination  https://github.com/OWNER/REPOSITORY.git (fetch)
> destination  https://github.com/OWNER/REPOSITORY.git (push)
```

Troubleshooting: Could not rename config section 'remote.[old name]' to 'remote.[new name]'

This error means that the old remote name you typed doesn't exist.

You can check which remotes currently exist with the `git remote -v` command:

```
$ git remote -v
# View existing remotes
> origin  https://github.com/OWNER/REPOSITORY.git (fetch)
> origin  https://github.com/OWNER/REPOSITORY.git (push)
```

Troubleshooting: Remote [new name] already exists

This error means that the remote name you want to use already exists. To solve this, either use a different remote name, or rename the original remote.

Removing a remote repository

Use the `git remote rm` command to remove a remote URL from your repository.

The `git remote rm` command takes one argument:

- A remote name, for example, `destination`

Example

These examples assume you're [cloning using HTTPS](#), which is recommended.

```
$ git remote -v
# View current remotes
> origin  https://github.com/OWNER/REPOSITORY.git (fetch)
> origin  https://github.com/OWNER/REPOSITORY.git (push)
> destination  https://github.com/FORKER/REPOSITORY.git (fetch)
> destination  https://github.com/FORKER/REPOSITORY.git (push)

$ git remote rm destination
# Remove remote
$ git remote -v
# Verify it's gone
> origin  https://github.com/OWNER/REPOSITORY.git (fetch)
> origin  https://github.com/OWNER/REPOSITORY.git (push)
```

> **Note**: `git remote rm` does not delete the remote repository from the server. It simply removes the remote and its references from your local repository.

Troubleshooting: Could not remove config section 'remote.[name]'

This error means that the remote you tried to delete doesn't exist:

```
$ git remote rm sofake
> error: Could not remove config section 'remote.sofake'
```

Check that you've correctly typed the remote name.

Further reading

- "[Working with Remotes](#)" from the *Pro Git* book