# **Remote Origin Already Exists Error Means and How To Fix It**

**"Remote Origin Already Exists" Error: Why Does It Happen?**

Picture this: You're following a Git tutorial you've found online. So far everything works fine. Then, you see a line similar to the following one:

git remote add origin <SOME-URL>/<SOME-REPOSITORY-NAME>.git

After trying to execute the command above, you get the infamous error message:

fatal: remote origin already exists.

Understanding this message is actually easy. Unlike centralized VCSs, Git doesn't have a central server. In Git, you can have what we call remote repositories, or simply remotes. Remotes represent repositories that you might have read and/or write access to. Those are usually on machines other than your own, and you access them via SSH or HTTP. Keep in mind that, despite the name, remotes aren't necessarily located on remote machines: despite sounding like an oxymoron, local remotes are totally possible.

Remotes have names to identify them, and you can have as many remotes per repository as you need or want. However, you can't have two remotes with the same name. So if you try to add a remote with the same name as an already existing remote, boom! Fatal error.

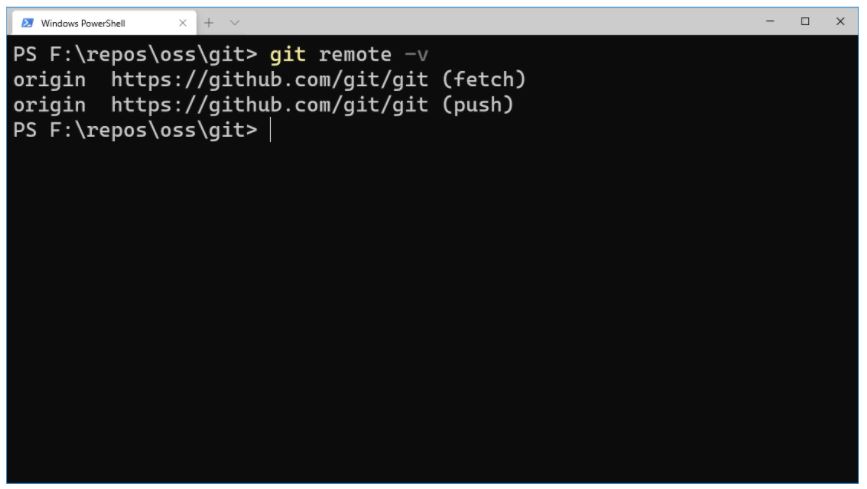
If you want to be really sure the remote called origin actually exists, you can easily check that by running the following command:

git remote

That will make your Git list all existing remotes for the current repository. If you want to get more detail, you can add the verbose parameter with the remote command, like this:

git remote -v

That will return not only the names of each remote but also its URLs:



By the way, the message will not always contain the actual word "origin." Let's say you're trying to add a remote called cloudbees but there's already a remote with that name. In this case, the error message would say:

fatal: remote cloudbees already exists.

Similarly to the way that the default branch in Git is called controller—though that could change in the near future—the default remote is called origin, but you could name it anything you like as long as it's a legal name in Git.

**Solving the "Remote Origin Already Exists" Error in Four Different Ways**

**1. Remove the Existing Remote**

The first scenario we'll cover is the one in which there's already a remote called origin, but it's the wrong remote for some reason. Let's say, for the sake of the example, that you used to use GitLab for storing your repositories online and then decided to switch over to GitHub (or vice versa). To go about that, you could follow the steps below:

Create a new repository online using GitHub or GitLab.

Go to your local repository and remove the existing origin remote.

Add the new online repository as the correct origin remote.

Push your code to the new origin.

If, for some reason, you skip step #2, that will cause Git to display the "remote origin already exists" message. So a possible solution here would be simply removing the existing remote:

git remote remove origin

As explained before, origin is just a name for a remote. It could be a different name for you. To make sure the remote is indeed deleted, you can use the Git remote command you saw earlier. Then, if everything is all right, you can go on to adding the desired remote.

**2. Update the Existing Remote's URL**

We've just shown you how to remove an existing remote, so you can hopefully add a new one, this time with the correct URL. However, you might be thinking that removing the remote and adding it again with a different link will have an eerily similar result as updating the URL of the existing remote. If that's the case, you're right, of course.

So let's see how to achieve the same result we got in the previous section but in a faster way. You just have to use a single command:

git remote set-url <REMOTE-NAME> <NEW-URL>

As we've said before, we keep talking about origin throughout this post, but there's nothing preventing you from working with whatever remote names you feel like. So a complete example with origin as the remote name and a URL to a real repo could look like this:

git remote set-url origin <https://github.com/git/git.git>

**3. Rename the Existing Remote**

Let's say that, for whatever reason, you already have a remote called origin. You want to add a new origin, but you also need to keep the old one. How would you go about it?

Easy. Rename the existing remote before adding the new one. Just run the following command and you're set:

git remote rename <old-name> <new-name>

So let's say you want to rename your origin remote to backup. You'd simply run:

git remote rename origin backup

Then you can add your new remote called origin normally, and you should no longer see the "remote origin already exists" error.

**4. Do Nothing!**

This is not a joke, I promise you. Here's the thing: Sometimes, you might get the "remote origin already exists" error when following a tutorial that has some step asking you to add a remote called origin. If you try to run the command and get the error message, it's possible that you've already executed that command and don't remember.

To check whether that's really the case, you can use the Git remote command with the verbose option, as we've covered before:

git remote -v

That will allow you to see the existing remotes along with the URLs they point to. If the existing remote already has the same URL provided by the tutorial, that means your repo is ready to go and you don't need to do anything else.