Working with remote Repositories

what is remote repositories ?
Remote repositories are **versions of your project that are hosted on the Internet or network somewhere**.

Adding Remote Repositories

We've mentioned and given some demonstrations of how the git clone command implicitly adds the origin remote for you. Here's how to add a new remote explicitly. To add a new remote Git repository as a shortname you can reference easily, run git remote add <shortname> <url>:

```
lab05-pc07@lab05pc07-H310M-S2-2-0:~/software1$ git remote add pb
https://github.com/Kazuto0p/software1.git

lab05-pc07@lab05pc07-H310M-S2-2-0:~/software1$ git remote -v
origin  git@github.com:Kazuto0p/software1.git (fetch)
origin  git@github.com:Kazuto0p/software1.git (push)
pb      https://github.com/Kazuto0p/software1.git (fetch)
pb      https://github.com/Kazuto0p/software1.git (push)
```

Inspecting a Remote

If you want to see more information about a particular remote, you can use the `git remote show <remote>`

```
lab05-pc07@lab05pc07-H310M-S2-2-0:~/software1$ git remote show
origin

lab05-pc07@lab05pc07-H310M-S2-2-0:~/software1$ git remote show origin
* remote origin
  Fetch URL: git@github.com:Kazuto0p/software1.git
  Push  URL: git@github.com:Kazuto0p/software1.git
  HEAD branch: main
  Remote branches:
    dev    tracked
    main   tracked
    master tracked
  Local branch configured for 'git pull':
    main merges with remote main
  Local ref configured for 'git push':
    main pushes to main (up to date)
```

Renaming and Removing Remotes

lab05-pc07@lab05pc07-H310M-S2-2-0:~/software1$ git remote rename pb paul
lab05-pc07@lab05pc07-H310M-S2-2-0:~/software1$ git remote
origin
paul

If you want to remove a remote for some reason — you've moved the server or are no longer using a particular mirror, or perhaps a contributor isn't contributing anymore — you can either use `git remote remove` or `git remote rm`:

```
$ git remote remove paul
$ git remote
origin
```

Fetching and Pulling from Your Remotes

```
$ git fetch <remote>
```

```
The command goes out to that remote project and pulls
down all the data from that remote project that you don't
have yet. After you do this, you should have references
to all the branches from that remote, which you can merge
in or inspect at any time.
```

If you clone a repository, the command automatically adds that remote repository under the name "origin". So, `git fetch origin` fetches any new work that has been pushed to that server since you cloned (or last fetched from) it. It's important to note that the `git fetch` command only downloads the data to your local repository — it doesn't automatically merge it with any of your work or modify what you're currently working on. You have to merge it manually into your work when you're ready.

OR

`git fetch` downloads the latest changes from the remote repository without modifying your current work, allowing you to check for updates without affecting your local files or branches.

Why it’s useful: It lets you see all the changes made by others before you decide to merge them into your work. It keeps your current work safe.

Pushing to Your Remotes

When you have your project at a point that you want to share, you have to push it upstream. The command for this is simple: `git push <remote> <branch>`. If you want to push your `master` branch to your `origin` server (again, cloning generally sets up both of those names for you automatically), then you can run this to push any commits you’ve done back up to the server:

```
$ git push origin master
```

This command works only if you cloned from a server to which you have write access and if nobody has pushed in the meantime. If you and someone else clone at the same time and they push upstream and then you push upstream, your push will rightly be rejected. You’ll have to fetch their work first and incorporate it into yours before you’ll be allowed to push.

OR

When you're ready to share your work with others, you use `git push` to send your changes to the remote repository. For example, to push your `master` branch to the remote called `origin`, you can use:

$ git push origin master

This command only works if you have permission to push to the remote, and if no one else has pushed changes since you last updated your copy. If someone else has pushed changes before you, your push will be rejected. In that case, you need to first fetch their changes and combine them with yours before you can push your updates.