Git @ Ohmori Group - Exercises 2

Giorgio Micaglio

Jul 25, 2025

1 Branching

Use the Git commands just learned during the lesson from the terminal to complete the following exercises:

1. Staying in the same Git repository, create a branch, for example called `new`. View the list of existing branches.
2. Switch to the new branch (checkout the new branch) and view the branch list again. What changed in the command output?
3. Create a new file and stage and commit it.
4. View the output of `git log`.
5. Go back to the main branch and look at the files present in the folder. The new file is not here!
6. To better see the differences between the two branches, use the command `git diff` followed by the name of the branch you want to compare with the one you're currently in. What differences appear?
7. Merge the two branches: remember to stay in the final branch where you want the other branch (which is no longer needed) to be merged.
8. If the `new` branch is no longer needed, delete it.

2 Merging

Use the Git commands just learned during the lesson from the terminal to complete the following exercises:

1. Initialize a Git repository in a new folder;
2. Create a text file, for example `hello.txt`, and using your preferred text editor add the line of text:

   ```
   Hello world!
   ```

   then stage and commit the file (all in the `main` branch);
3. Create a new branch, for example call it `new`, and modify the text file `hello.txt` as follows:

   ```
   Hello world!  How are you?
   ```

   then stage and commit the file in the `new` branch.
4. Switch to the `main` branch and again modify the file `hello.txt` with the text:

   ```
   Hello world!  The sun is shining outside!
   ```

   then stage and commit the file in the `main` branch;
5. Now it's time! Merge the `new` branch. If everything went as expected, a conflict should have arisen on the file `hello.txt`. Try running the `git status` command and then resolve the conflict;
6. Once the conflict is resolved, stage and commit the file.

3 Remote Work

Here's a partner exercise! Your first task is to find a partner — you'll be nicknamed Alice and Bob!

Use the Git commands just learned during the lesson from the terminal to complete the following exercises:

1. Alice must create a repository on her GitHub account and add Bob as a collaborator (don't worry, we'll show you how!). When creating the repository, allow GitHub to generate the README.md file.
2. Both of you clone the repository to your computers. You should both start with the README.md file on your local machines.
3. Alice edits the README.md file with an editor of her choice, and after staging and committing her changes, she pushes them to the online repository.
4. Bob also edits the README.md file and after staging and committing his changes, tries to push them to the online repository. How did it go?
5. Git should not have allowed the last command because the online repository was one commit ahead of the local one! From the lesson, we learned what to do in such situations! Use the `git fetch`, `git merge`, and `git push` commands to integrate Bob's changes into the online repository.
6. Alice creates a new branch and modifies the README.md file again, saving the changes in the new branch. Now Alice wants to merge her new branch and push it to the online repository.
7. From the lesson we learned that to do this we must follow this sequence:
   - fetch;
   - fast-forward merge of the local `master` branch with the new local branch;
   - 3-way merge of the local `master` branch with the fetched branch (`origin/master`);
   - push to the online repository.