Git for CNaaS-NMS

Git is a version-control system for source-code or other text-files. In the context of CNaaS-NMS we use git to manage text-files containing configuration-templates and settings. Git has many features but to manage these templates and settings there are only a few things we have to learn. Git will allow us to synchronize these text files between all the users in the project and the central API software. In addition git will keep a detailed log of all changes that has been made and allow us to restore any previous version of the files.

First some git concepts:

Repository: All files concerning a specific project is contained in a git repository. For CNaaS, there might be a repository for all settings files for your CNaaS lab network for example.

Remote repository: A remote server that all users can synchronize their changes to and from. This might be a server in the SUNET cloud infrastructure or some public cloud like GitHub.

Local repository: A copy of all the files from the remote repository saved on your local computer.

VSCode - Visual Studio Code

Install visual studio code and git on your computer.

For windows users: enable ssh-agent and add your ssh-key

Ctrl-Shift-P Git: Clone Enter URL to git repository

Commandline: First use of existing remote repository

Open up a terminal and change to a directory where your want to keep all your git repositories.

```
cd ~/git
```

Clone (copy) a remote repository:

```
git clone git@git.myorg.com:cnaas-lab-settings.git
```

This will copy the specified remote repository files to your local computer, you can find the contents in the directory cnaas-lab-settings in this example:

```
cd cnaas-lab-settings/
```

Working with a repository that you cloned

Open a terminal and go to the repository:

```
cd ~/git/cnaas-lab-settings
```

To make sure you have all the latest changes that other users might have making to the repository since you last checked you have to "pull" these changes from the remote:

```
git pull
```

You can then edit any files in the repository with your favorite text editor. When you have made your changes and saved the files you can check the current status:

```
git status
```

This will tell you if you have some "changes to commit" and list the filenames of any changed files. To save these changes to your local change history in git you have to commit the change using this command and enter a message describing your work:

```
git commit -a -m "added new vxlan for students in building C"
```

The changes have now been saved in your local git repository, but to make the changes available to other users or the central API you need to push your changes to the remote repository:

```
git push
```

You can use "git status" between each step here to see what you need to do next.

Video explanation

You can find a short 5 min video explaining almost the same concepts here:

However in that video there is an assumption that only one user is working with the repository and therefore he does not explain the "git pull" command. If there are multiple users of the repository you always have to remember to use "git pull" before you start editing any files to avoid "merge conflicts" (when multiple users changed the same thing unaware of each other).

External links

Git branching: https://learngitbranching.js.org/