
Week 1 — *Using GIT*

The goal of the present exercise is to use the revision control GIT.

Exercise 1: *Installing a GIT client*

1. Under Debian/Ubuntu/Mint you should hit:

```
sudo apt-get install git
```

2. Open a console. You will now need to give a name and email address to GIT, so that it knows who is committing.

```
git config --global user.email "your.email@epfl.ch"  
git config --global user.name "Your Name"
```

Exercise 2: *Creating your own c4science-GIT repository*

1. To create your own repository go to <https://c4science.ch>
2. Log in using your EPFL credentials
3. Click on the link *Create a new repository* and follow the instruction to create your first repository named `<name-of-your-choice>`. In the “Tags” field, add the tag “Scientific Programming For Engineers - Students”.
4. In order to activate the repository, click on the link *Actions→Manage Repository*. Under option *Basics*, click on the *Actions→Activate Repository*.
5. Open a terminal.
6. In order to be able to retrieve repositories on c4science, you will need a pair of `ssh` keys. Run in the console:

```
ssh-keygen # Follow the steps  
cat .ssh/id_rsa.pub
```

Copy the output of the last command. Go to <https://c4science.ch/settings>. On the left panel, click “SSH Public Keys”, then “SSH Key Actions”, “Upload Public Key”. Give it a name and paste the output of the `cat` command.

7. C4Science should provide you with an URL for your repository (it starts with `ssh://`). You can clone your repository

```
git clone <c4science-repository-url>
```

You now have a local clone of your EPFL repository named `<name-of-your-choice>` (a new directory should have been created for it). You can go in the directory using:

```
cd <name-of-your-choice>
```

8. Inside this directory, create a new file `test.cpp`.

10. You can finish the conflict resolution by comiting the solution

```
git commit -am "resolution message"
```

11. And finally you can push your changes

12. What does the '-a' stands for ? You can find out by issuing the command:

```
git help commit
```

You can thus realize that all commands are documented within the command line:

```
git help the-command-I-am-intrigued-about
```

Exercise 4: Add a remote to the class repository

In the repository <name-of-your-choice>, add a new remote that points to the official SP4E repository.

```
git remote add upstream ssh://git@c4science.ch/source/sp4e.git
```

You can see all the remotes in you repository with:

```
git remote -v
```

Pull the class material in your repository:

```
git pull upstream master
```

You will probably get a screen with a merge message. If it is VIM, press `:wq<Enter>` to save and exit. You should now be able to find this week's exercises. If you see an error about 'unrelated histories' you should use the following instead:

```
git pull --allow-unrelated-histories upstream master
```

You can push this new content to your personal repository:

```
git push
```

All material for the class, including class notes and exercices starters will be provided through this GIT repository. Avoid putting modifications directly in the class material, or it is very likely that you will get conflicts if you try to pull from `upstream`. Instead, copy the files you need to modify elsewhere in your repository.

Exercise 5: Branches

You can find an interactive exercise on <http://learngitbranching.js.org/>.