Software Carpentry 14-17 April, 2020

Day 3

Code of Conduct

Participants are expected to follow our code of conduct:
https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

License

All content is publicly available under the Creative Commons Attribution License:
https://creativecommons.org/licenses/by/4.0/

This Document:

https://tinyurl.com/2014-04-14-swc-day3

Master Document:

https://edu.nl/tcyt9

Workshop website:

https://escience-academy.github.io/2020-04-14-SWC-online/

Instructor:

Pablo Rodríguez Sánchez

Helpers:

Evert Rol, Sarah Alidoost, Jaro Camphuijsen, Johan Hidding

Agenda:

Schedule on the workshop website

09:00 --- Good morning coffe & testing setup ---

09:30 What is version control and why should I use it
09:35 Setting up git
09:40 Creating a repository
09:50 Tracking changes
10:10 Exploring history
10:30 Ignoring things
10:35 Remotes in GitHub

11:00 --- Morning break ---

11:30 Collaborating
12:00 Conflicts

12:30 --- Wrap-up ---

Roll Call:

name / affiliation / pronouns (optional) / contact, social media etc. (eg. twitter) / background (formal trainining)

- Pablo Rodríguez-Sánchez / eScienceCenter / TW: @DonMostrenco / pabrod.github.io / physicist / mathematician
- Sarah Alidoost / Netherlands eScience Center / she, her / f.alidoost@esciencecenter.nl / (geo)statistician / Earth Observation Science
- Mateusz Kuzak / the Netherlands eScience Center / he, him / t: @matkuzak
- Terezinha Souza/Maastricht University/t.souza@maastrichtuniversity.nl/Post-doc in life sciences
- Elisabeth Heijmans/ Leiden University/she her/ e.a.r.heijmans@hum.leideuniv.nl / Post-doc in History
- Banafsheh Abdollahi/ TU Delft/ b.abdollahi@tudelft.nl

- Jaro Camphuijsen / Netherlands eScience Center / he, him / j.camphuijsen@esciencecenter.nl
- Marco Dal Molin / EAWAG, Zurich (CH) / marco.dalmolin@eawag.ch / Environmental Engineer, Hydrologist
- Alessia Vitiello/ Wageningen University & Research, Postdoc at Laboratory of Entomology/ alessia.vitiello@wur.nl/ Plant biotechnology
- Victor Bernal / University of Groningen/ v.a.bernal.arzola@rug.nl/ victor.arturo.bernal@gmail.com/ Physics, Maths, Genomics
- Michela Busana/ University of Groningen/ michebusana@gmail.com / Biology, Ecology
- Baharak Hooshiar /Helmholtz Centrum Munich //baharak.hooshiar@helmholtz-muenchen.de / Biologist
- Bruna Vieira / Radboudumc / bruna.dossantosvieira@radboudumc.nl / Public health
- Evert Rol / Netherlands eScience Center / e.rol@esciencecenter / Astrophysics
- Johan Hidding / Netherlands eScience Center / j.hidding@esciencecenter.nl / Astrophysics
- Julien Dupeyroux / TU Delft / j.j.g.dupeyroux@tudelft.nl / Postdoc Aerospace Engineering
- Hossein Asadi Kalameh / ULG / asadi.hosein@gmail.com/ PhD student in Mechanical engineering
- Paula Martinez Lavanchy/ TU Delft/p.m.martinezlavanchy@tudelft.nl/ Research Data Officer

Icebreaker

We'll create 5 videocall rooms with 4-5 people, and we'll introduce each other. Just as in a regular meeting.

Contents:

- to check if git is installed:

```
git --help
```

- to configure it

```
git config --global user.name <your name>
git config --global user.email <your email>
```

- to check the configuration

```
git config --list
```

- to check the status of your repository

```
git status
```

- to initialize a repository

```
git init
```

- to check files inside the repository

```
ls -a
```

- to make a directory (folder)

```
mkdir <folder name>
 cd <folder name>
```

example: cd planets

- to make a file

```
nano <file name.txt>
```

example nano mars.txt

to save and exit (ctrl+o and then ctrl+x)

- to add a file to git history

```
git status
```

it shows the changes in red color

```
git add <folder name>/<file name.txt>
```

example: git add planets/mars.txt

- to commit changes

```
git commit -m “<your message>”
```

example: git add -m “Add information about Mars”

- to see information

```
git status
```

- to see git history (a list of commits with Author name, date, and message)

```
git log
```

- let’s make another file

```
cd planets
nano <your file name>
```

example: nano mercury.txt
write something, for example “no moon”
to save and exit (ctrl+o and then ctrl+x)

- to see inside the file

```
cat <your filename.txt>
```

example: cat mercury.txt

- let’s add more files

```
nano earth.txt
```

write something, for example “1 moon: The moon”
to save and exit (ctrl+o and then ctrl+x)

- to add changes (adding two files at the same time)

```
git add earth.txt mercury.txt
```

- to see files that changed

```
git status
```

the changed files in green color

- to see inside the file

```
cat <your filename.txt>
```

example: cat earth.txt, it will show: “1 moon: The moon”

- to write a message (longer than one line message)

```
git commit
```

it opens an editor, you can write your message there
** it opens the default editor, in **nano** to save and exit (ctrl+o and then ctrl+x)**
** it opens the default editor, in **vim** to save and exit (esc + :wq)**

- to see git history (a list of commits with author name, date, and message)

```
git log
```

- let’s make another file

```
nano mars.txt
```

3 moons:
Phobos
Deimos
Terror
** it opens the default editor, in **nano** to save and exit (ctrl+o and then ctrl+x)**
** it opens the default editor, in **vim** to save and exit (esc + :wq)**

- to see files that changed

```
git status
```

the changed files in green color

- to see what the changes are

```
git diff
```

deleted information in red colors
new information in green color

- to add changes

```
git add mars.txt
git commit -m “Update information about Mars
Astonishing news: a new moon for Mars has been discovered
```

- to see git history (a list of commits with author name, date, and message)

```
git log
```

- to see only the title of the commit

```
git log --oneline
```

- to know more about git log

```
git log --help
```

As an example, to see a graphical representation of git history

```
git log --oneline --graph
```

- to see only the title of the commit

```
git log --oneline
```

**now you can see three commits there”

- to see differences between the current version and one/two versions before

```
git diff HEAD~1 mars.txt

git diff HEAD~1 HEAD mars.txt

git diff HEAD~2 HEAD~1 mars.txt

git diff HEAD~2 HEAD~1
```

- to see the changes in another version

```
git log --oneline
```

** it shows a list of commits**

```
git checkout <commit id> mars.txt
git status
cat mars.txt
```

- to back to the current version

```
git checkout HEAD mars.txt
git status
```

(Coffee time)

- to ignore file or files

use `.gitignore` text file

- let’s open a browser and go to https://github.com/

log in with your account
Next to your profile photo, there is a **+**
Click on **+** and then **New repository**
in Repository name, you can type a name for example planetsexercise
choose public

click on **Create repository**

- let's go to Terminal (command console)

```
git status
git log --online
git remote -v
```

- to add a remote

```
git remote add origin https://github.com/.../planetsexercise.git
```

** this command line and the link is shown in the browser**

```
git remote -v
git push -u origin master
```

- *Another approach to add a remote*

```
git remote set-url origin git@github.com: <your
username>/planetsexercise.git
git push -u origin master
```

- let's go to the browser

we can see the folder "planets" and files in the repository.

- let's go to Terminal (command console)

```
cd planets/
```

- lets make another file and write in it

```
nano venus.txt
```

0 moons

** it opens the default editor, in **nano** to save and exit (ctrl+o and then ctrl+x)**

** it opens the default editor, in **vim** to save and exit (esc + :wq)**

```
git status
```

** it shows the venus.txt in red color**

```
git add venus.txt
git status
```

** it shows the venus.txt in green color**

```
git commit -m "Add information about Venus"
```

- to push the new changes to our repository in github

```
git push
```

- let's go to the browser (and refresh it)

we can see the file "venus.txt" in the folder "planets" .

to access the information in the repository with clone

- let's go to Terminal (command console) and make a new folder

```
cd ..
cd ..
mkdir clonex
cd clonex
ls -a
```
** we see no files here**

in the browser, click on **clone or download**, use **HTTPS**, copy the URL"

- in the Terminal, paste the URL after `git clone` as:

```
git clone https://github.com/.../planetsexercise.git
ls -a
cd planetsexercise/
ls -a
```

- we want to see the list of branches and make a new one

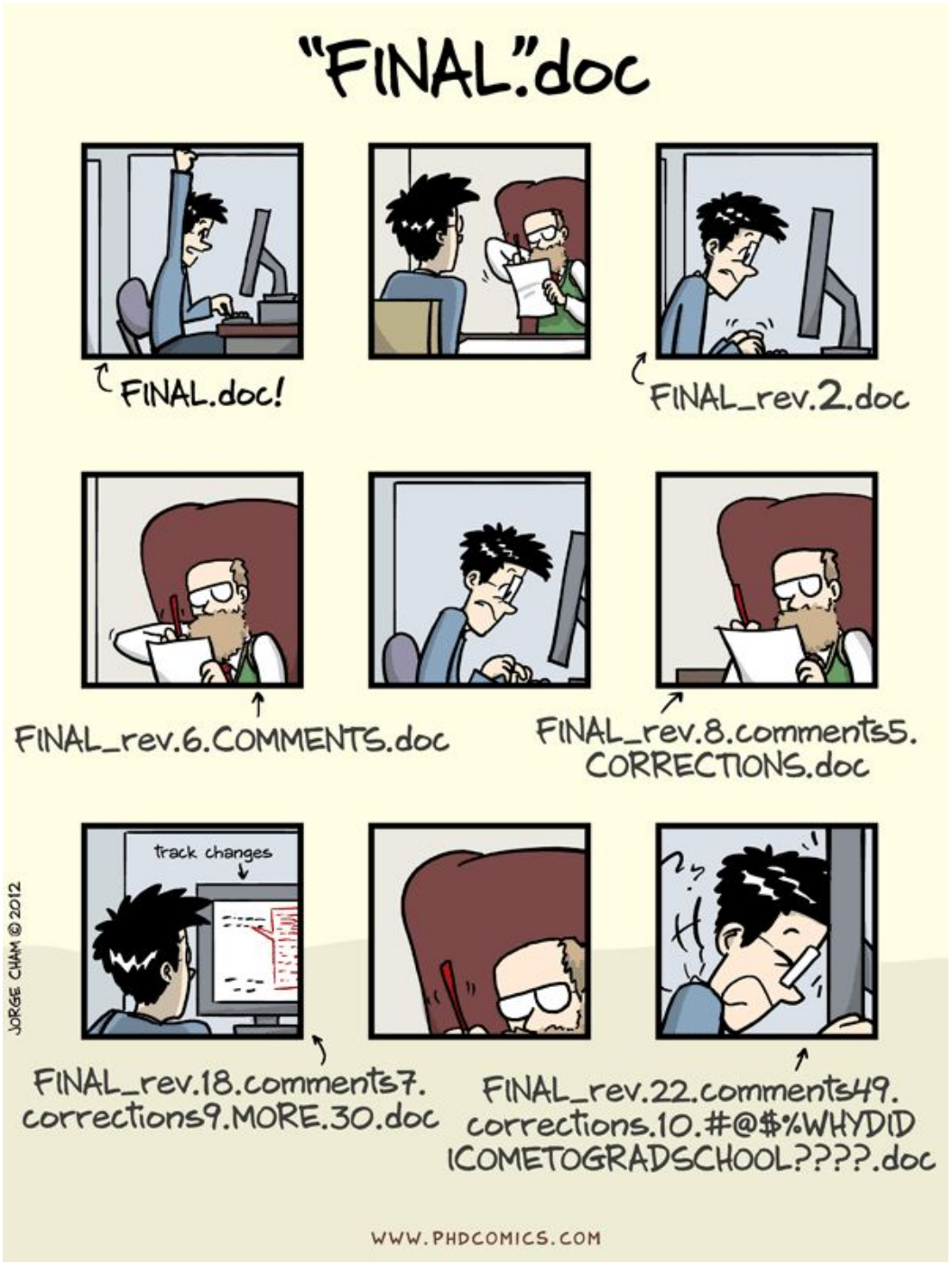
in the browser, click on **branch:master**
it shows a list of branches in the repository (for now only master)
We can create a branch like feature/outer

lets select **master** branch and click on create a file.
We choose a name like Readme.md and then commit.

- we want to see the Readme.md locally, so in the terminal

```
ls
git pull
ls
```
now we see the file readme.md

(Finish, Q&A)



Feedback

what went well / what you liked / favourite new things learned

Very interesting to learn about git and github as tools/ I very much like the “command history‘ in the document/

Very patient helpers, thank you for answering all the questions so fast.

I was very satisfied by today’s tutorial on Git. This was clear and following at very good pace.

Very useful topic

Very interesting topic and very well explained

what should be improved / what was not clear enough

Suggestion: it would be nice, if possible, to add an extra morning to finish to cover all the points, or at least to have more insight about branches and collaboration on GitHub

I would have liked the tutorial to go up to merging repositories, from the original to a forked one.

the pace can be slightly faster