Welcome to the Carpentries Etherpad !

dsc.uva.nl

Software Carpentry Edubadge

Request your badge

- Create an eduID on https://eduid.nl/.
- Use your personal email address, not your institutional email address! This will ensure that you can access your Edubadges along your future career steps.
- Go to e://www.edubadges.nl/public/92rVIhSyQs2HjthedJIg0g to enroll for the Software Carpentry badge.
- Click on 'Request'.

Once we have received your request, we will evaluate it and approve it if you have completed all course requirements. This includes checking your presence throughout the workshop.

I will share instructions on how to upload it in Hora Finita

in a nutshell:

Once your badge is approved, you can proceed as follows to register it in Hora Finita.

1. Log in to edubadges.nl and go to ‘Open je backpack/Open your backpack’
2. Select the badge for Software Carpentry
3. Click on the button ‘Deel/Share’

4. Copy the link
5. Paste the link into your browser. The link will lead to the public page of your badge. Make a screenshot of this public page and be ready to copy the link to Hora Finita as well.
6. Open a new training activity registration form in Hora Finita, provide the link to your badge in the field ‘Omschrijving/Description’ and upload your screenshot below ‘Bewijs van deelname documenten/Proof of attendance documents’

Post-workshop survey:

Make you sure you have filled the Software Carpentry workshop pre-survey so we know what is your pre-knowledge before the workshop:

Post-workshop survey: https://carpentries.typeform.com/to/UgVdRQ?slug=2023-05-30-uvavu

Day 4 : Git & GitHub

Please add your name to the list of participants every day. This will help us to get credit points for your participation from the graduate schools.

Lesson Material: https://swcarpentry.github.io/git-novice/

Instructor

Koen Leuveld,Data Steward FSW

Helpers

Thomas Pronk, Research Software Consultant, Amsterdam UMC

Link to SSH setup: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

Learners:

- Tycho Hofstra, UBVU
- Gideon

- Pascal
- Elja
- Giovanni
- Ira
- Job
- Robert
- Evgenia
- Aleksandra
- Tori
- Steven
- Xihan
- Thijmen
- Xiaoyu
- Jorik
- Boris
- Polina

Git

```
 1  cd Desktop
 2  cd planets
 3  ls
 4  git log
 5  nano mars.txt
 6  git diff HEAD mars.txt
 7  git diff HEAD~1 mars.txt
 8  git diff HEAD~2 mars.txt
 9  git diff mars.txt
10  ls -a
11  git log --oneline
12  git diff 4fce25d mars.txt
13  git diff HEAD~2 mars.txt
14  git log
15  git status
16  git restore mars.txt
17  cat mars.txt
18  git log
19  git log --oneline
20  git restore -s 4fce25d mars.txt
21  cat mars.txt
22  git status
23  git diff mars.txy
24  git diff mars.txt
25  git restore mars.txt
26  git status
27  git log --oneline
```

```
28  git log
29  git log --oneline
30  git log mars.txt --oneline
31  git log --oneline mars.txt
32  git log --oneline
33  git restore 4fce25d mars.txt
34  git restore -s 4fce25d mars.txt
35  git diff mars.txt
36  git status
37  git add mars.txt
38  git status
39  git restore --staged mars.txt
40  git status
41  git restore mars.txt
42  git status
43  cat mars.txt
44  git log
45  git log --oneline
46  echo "Line 1" > venus.txt
47  git add venus.txt
48  echo "Line 2" >> venus.txt
49  git commit -m "Comment on Venus"
50  git restore venus.txt
51  cat venus.txt
52  git log --oneline
53  git restore -s 8a0d27b venus.txt
54  ls
55  git status
56  restore venus.txt
57  git restore venus.txt
58  mkdir results
59  touch a.dat b.dat c.dat results/a.out results/b.out
60  ls
61  cd results
62  ls
63  cd ..
64  git status
65  nano .gitignore
66  git status
67  git add .gitignore
68  git commit -m "Ignore data files and results folder"
69  git log --oneline
70  git status
71  git status --ignored
72  nano .gitignore
73  cd ..
74  cd ,,
75  cd ..
76  cd git
```

```
   77  ls
   78  cd thesis
   79  git log --oneline
   80  git log
   81  history
   82  history
   83  cd ..
   84  cd ..
   85  cd Desktop/
   86  cd planets
   87  history > history.txt
   88  git log
   89  git status
   90  history > history.txt
       git clone https://github.com/kleuveld/planets.git ~/Desktop/vlad-planets
       git clone https://github.com/kleuveld/planets.git ~/Desktop/wolfman-planets
    2  cd vlad-planets/
    3  git log --oneline
    4  nano pluto.txt
    5  git add pluto.txt
    6  git commit -m "Add notes about pluto"
    7  git log --oneline --all
    8  git push origin main
    9  git log --oneline --all
   10  history
```

Excercises:
Which commands below willlet you recover the last committed version of the Python script called data_cruncher.py?
1. $ git restore HEAD
2. $ git restore data_cruncher.py
3. $ git restore -s HEAD~1 data_cruncher.py
4. $ git restore -s <unique ID of last commit> data_cruncher.py

Write your answer(s) here:
4
2

Authentication on Mac with SSH:
git remote remove origin
https://swcarpentry.github.io/git-novice/instructor/07-github.html

Feedback for the instructor:

What I like about Git
Its a good fundermental for me to build on

Having a structured version control
Conceptually interesting.
it seems you can apply many parameters _> flexibility
command line may be a really quick, once you are used to it.
flexibility in the workflow and integration with other tools
Ability to cooperate and share (plain text) data
Don't need to worry about losing files

What I don't like about Git
not too basic for biginners
Not automatically commiting
Not always intuitive, even GUI not.
Currently don't see relevance for my way of working (doc/excel etc) --> perpahs first need to know latex?

- (You could start writing in Markdown or just notepad for text files, and save your excel files as csv)

I keep forgetting the commands and the order they need to be entered

I need to have all my coauthors on boat

Am I going to use Git (and why?)
Not immediately, as most of my work is with MS Word and Excel
I am using Git as kind of personal back-up service, but I will need to commit more often to make full advantage of the version control
Probably not - do not find it intuitive yet --> more training first?
In order for it to work, I'll need to commit more often
Can I use it to store the versions of WORD or EXCEL

Tip for this workshop:
Elaborate on latex first, before going into git

Top of this workshop:
Touched upon a wide variety of related topics (python, git, shell etc). However, on everyone of those you woudl require substantially more training to start using.
The use of examples to do it yourself rather than just copying instructions

Speed check, please vote by writing + (plus) sign below
go-faster:
go-slower:
okay:

Questions:

- (TP) Could you use **git restore -s HEAD *** (with a wildcard instead of a filename)? (Update: tried it out, it works)

Bytes & Bites (June 6, 16.00 - 18.00):

Register here: https://vu-nl.libcal.com/calendar/universitylibrary/bytesbites

In about two weeks, we have a special treat for you during the Bytes&Bites meeting: Douwe Molenaar will talk about data wrangling. What you should think about and what can go wrong!
We very much hope you will be there! The most important things are listed below.

Date 6 June
Time 16:00 - 18:00 (come whenever you want and stay as long you wish)
Location: O2 Overlegkamer 01W08 (Vrije Universiteit Amsterdam)
Register before, so that we know how much pizza to buy. No worries if you can't be there the whole period, everyone is welcome anytime 😉 And please share this invite with your colleagues.

Similar to our previous editions, at the B&B we invite everyone interested in programming tools and computational research to build skills and community. Folks of all experience levels are welcome to learn and collaborate with one another (and eat pizza). Do you want to improve your programming skills, ask questions related to programming, and feel yourself part of the broader VU community working on their programming skills? Come to Bytes and Bitespeer support for researchers and students learning to program!

Day 3 : Shell + Git

Please add your name to the list of participants every day. This will help us to get credit points for your participation from the graduate schools.

Learners:

- Elisa Rodenburg, UBVU
- Aleksandra
- Giovanni
- Robert
- Ira
- Xiaoyu
- Gideon
- Elja

- Job
- Jorik
- Pascal
- Tori
- Thijmen
- Evgenia
- Steven
- Xihan
- boris
- Polina

Shell

Lesson Material: https://swcarpentry.github.io/shell-novice/
Data：https://swcarpentry.github.io/shell-novice/data/shell-lesson-data.zip
Instructions for keeping track of history in the second terminal window:
https://github.com/4TUResearchData-Carpentries/documentation/blob/master/command-history.md (you can simply copy these commands without really needing to understand it)

Commands used so far:

```
ls #for listing what's in a folder
pwd # this is the command to print the working directory
ls -F
clear
man ls
ls --help
pwd
ls -F Desktop
cd Desktop
pwd
cd shell-lesson-data
pwd
cd exercise-data
pwd
cd Desktop/shell-lesson-data/exercise-data
ls -F
cd shell-lesson-data
cd ..
pwd
ls -Fa #you can string together different options; as many as you like
cd
pwd
cd - #to go back to the directory where you previously were
cd ~ #this is a tilde: find it next to the 1
cd Desktop/shell-lesson-data/exercise-data
```

```
ls -F
mkdir thesis
ls -F
mkdir ../project/data ../project/results #THIS IS NOT CORRECT
mkdir -p ../project/data ../project/results
ls -FR ../project
cd thesis
nano draft.txt
ls -F
cat draft.txt
cd ..
mv thesis/draft.txt thesis/quotes.txt
ls -FR thesis
mv -i thesis/quotes.txt .
ls -F
cp quotes.txt thesis/quotations.txt
ls -F . thesis
cp -r thesis thesis_backup
ls -F
ls -R thesis_backup/
rm -i quotes.txt
ls -F
rm -i thesis #THIS ALSO THROWS AN ERROR
rm -ir thesis
# wildcards stand for one or more characters
# * asterisk stands for 0 or more characters
# ? matches exactly 1 character
cd proteins
pwd
ls
wc cubane.pdb
wc *.pdb
wc -l *.pdb
# use wc -w for the number of words and wc -m for the number of characters
wc -lw *.pdb
wc -l
wc -l *.pdb > lengths.txt #the greater-than sign writes the output of this command to the file lengths.txt
ls lengths.txt
cat lengths.txt
can methane.pdb
cat methane.pdb
cat lengths.txt
cd ..
cat numbers.txt
sort numbers.txt
sort -n numbers.txt
cd proteins #alkanes
cat lengths.txt
sort -n lengths.txt
sort -n lengths.txt > sorted-lengths.txt
ls
cat sorted-lengths.txt
head -n 1 sorted-lengths.txt
# we can combine commands with a pipe |
# a pipe is above the return key, use Shift
sort -n lengths.txt | head -n 1
wc -l *.pdb | sort -n | head -n 1
# > writes to a file
# >> appends to a file
echo prints text
echo My name is Elisa
echo My first name is Elisa > about-me.txt
cat about-me.txt
echo I work in the University Library >> about-me.txt
cat about-me.txt
history | tail -n 15 > command-history.txt #save the last 15 lines of your command history to a file called command-history
cat command-history.txt
```

Git

Commands used:

```
git
# we will configure our name and email address
# git config --global user.name "Elisa-on-GitHub"
git config --list
# git config --global user.email "github email"
# git config --global core.editor "nano -w"
# git config --global init.defaultBranch main
git config --list
cd
cd Desktop
mkdir planets
cd planets
git init
ls -F
ls -Fa
git status
nano mars.txt
ls -F
cat mars.txt
git status
git add mars.txt
git status
git commit -m "Start notes on Mars as a base"
git status
git log
nano mars.txt
```

```
cat mars.txt
git status
git diff
git commit -m "Add concerns about effects of Mars' moons on Wolfman"
git add mars.txt
git commit -m "Add concerns about effects of Mars' moons on Wolfman"
nano mars.txt
cat mars.txt
git diff
git add mars.txt
git diff
git diff --staged
git commit -m "Discuss concerns about Mars' climate for Mummy"
git status
git log
git log -2 # specify how many previous commits you'd like to see
git log --oneline
git log --oneline --graph
mkdir spaceships
touch spaceships/apollo-11 spaceships/sputnik-1
git status
git add spaceships
git commit -m "Add some initial thoughts about spaceships"
```

Lesson Material: https://swcarpentry.github.io/git-novice/
No data required :)

Feedback for the instructor:

Speed check, please vote by writing + (plus) sign below
go-faster: ++
go-slower:
okay:

Questions:

You mentioned you created a script for the university to determine contents of (sub)folders/archive. Would it be possible to share this script? I may want to use this to help clean up my own PC...

- Elisa will share the script here!

How can I add and commit at the same time in one command:

- git -help gives you more insights in how to use the commands
- git commit -am "Your commit message"

- or use an alias: git config --global alias.add-commit '!git add -A && git commit'
    - `git add-commit -m 'My commit message'`

I made a tyo - How can I change the commit message?

- git commit --amend

Day 2 : Python (Part 2)

Please add your name to the list of participants every day. This will help us to get credit points for your participation from the graduate schools.

Learners:

- Max Paulus, UBVU
- Tori
- Jorik
- Job
- Vedika
- Gideon
- Elja
- Aleksandra
- Evgenia
- Robert
- Ira
- Giovanni
- Thijmen
- Steven
- Xiaoyu
- Pascal
- Boris
- Polina

Python part II

Lesson Material:
Live Code: https://github.com/ubvu/python-ecology-05-23
Data：https://datacarpentry.org/python-ecology-lesson/index.html#data
Exercises/Slides:
https://docs.google.com/presentation/d/1O8MAzat_9aFByK2Oyu5497VdryQpNtJ_suvEGTQ2kx8/edit?usp=sharing

Feedback for the instructor:

Speed check, please vote by writing + (plus) sign below
go-faster: +++++
go-slower: ++
okay:+

Questions:

Pascal: I would like to change the (default) browser that Jupyter uses to Chrome (currently it opens in Edge). How can I do that? -

- by default, jupyter should use the default browser when opening, so that appears to be Edge in your case; I would first try to change your default browser if you don't want to use Edge; otherwise there is an option to modify the config file; it is mentioned in the link you shared, the parameter is c.NotebookApp.browser; you can point it to the executable of your browser; for me, it only works if you do it like this: "'C:/Program Files/Google/Chrome/Application/chrome.exe' %s" (notice the double quotes around the entire expression and the single quotes around the path

How do you get autocomplete in notebooks?

- its a function in jupyter notebooks, so it will only work here
- use the TAB key

This website contains useful visualizations for the different types of joins:
https://blog.codinghorror.com/a-visual-explanation-of-sql-joins/

pass is also used by programmers before you actually start writing a loop or function, as a kind of placeholder, i.e. the function runs but it doesn't do anything yet

range() returns a so called **generator** which is different from a list as the next value of the sequence is only generated when we specifically tell it to; calling list() on the range() will compute/generate all the values at once

Day 1 : Python (Part 1)

Please add your name to the list of participants every day. This will help us to get credit points for your participation from the graduate schools.

Learners:

Job
Pascal

Steven
Meta
Robert
Ira
Evgenia
Tori
Giovanni
Vedika
Thijmen
Jorik
Xiaoyu
Aleksandra
Xihan
Elja
Polina
harda
Boris
Evgenia
Gideon

Python part I

Lesson Material:

Live Code: https://github.com/ubvu/python-ecology-05-23
Data：https://datacarpentry.org/python-ecology-lesson/index.html#data
Exercises/Slides:
https://docs.google.com/presentation/d/1O8MAzat_9aFByK2Oyu5497VdryQpNtJ_suvEGTQ2kx8/edit?usp=sharing

Feedback for the instructor:

Speed check, please vote by writing + (plus) sign below
go-faster: ++++
go-slower:
okay: ++

Notes & Questions?

Markdown has a lot of possibilities, just look at this cheat sheet: https://www.markdownguide.org/cheat-sheet/

When using a variable that was previously assigned to iterate over something in a for loop, that variable will be overwritten by that for loop. Functions have internal variables that do not affect 'global' variables. That said, it is a good idea to make sure you use unique variable names that are descriptive and do not overwrite other variable names, or standard function names (for example print, etc.).

Dot notation for selecting columns does not work if the column name contains a whitespace, like **species id**, or when the name refers to an existing function, e.g. count, **so I recommend to always use the [] notation when you're unsure**