

# Intro: Files, BASH and Git + Markdown

---

Ádám T. Kocsis

Friedrich-Alexander-Universität Erlangen-Nürnberg

2022-08-22



# Why?

and GitHub

# Paleontological data in the 21<sup>st</sup> century

We have gone a long way...



Collectors only  
-1960s



Pioneers  
1960-1990/2000



Community of  
database-based research

# Being FAIR

A standard way to publish data and data-based research.

- Findable
- Accessible
- Interoperable
- Reproducible



[www.go-fair.org](http://www.go-fair.org)

## scientific **data**

[Explore content](#) ▾ [About the journal](#) ▾ [Publish with us](#) ▾

[nature](#) > [scientific data](#) > [comment](#) > article


[Open Access](#) | [Published: 15 March 2016](#)

### The FAIR Guiding Principles for scientific data management and stewardship

[Mark D. Wilkinson](#), [Michel Dumontier](#), ... [Barend Mons](#)  [+ Show authors](#)

[Scientific Data](#) **3**, Article number: 160018 (2016) | [Cite this article](#)

**474k** Accesses | **4409** Citations | **2001** Altmetric | [Metrics](#)

 An [Addendum](#) to this article was published on 19 March 2019

# Reproducibility

## The foundation of the scientific experiment

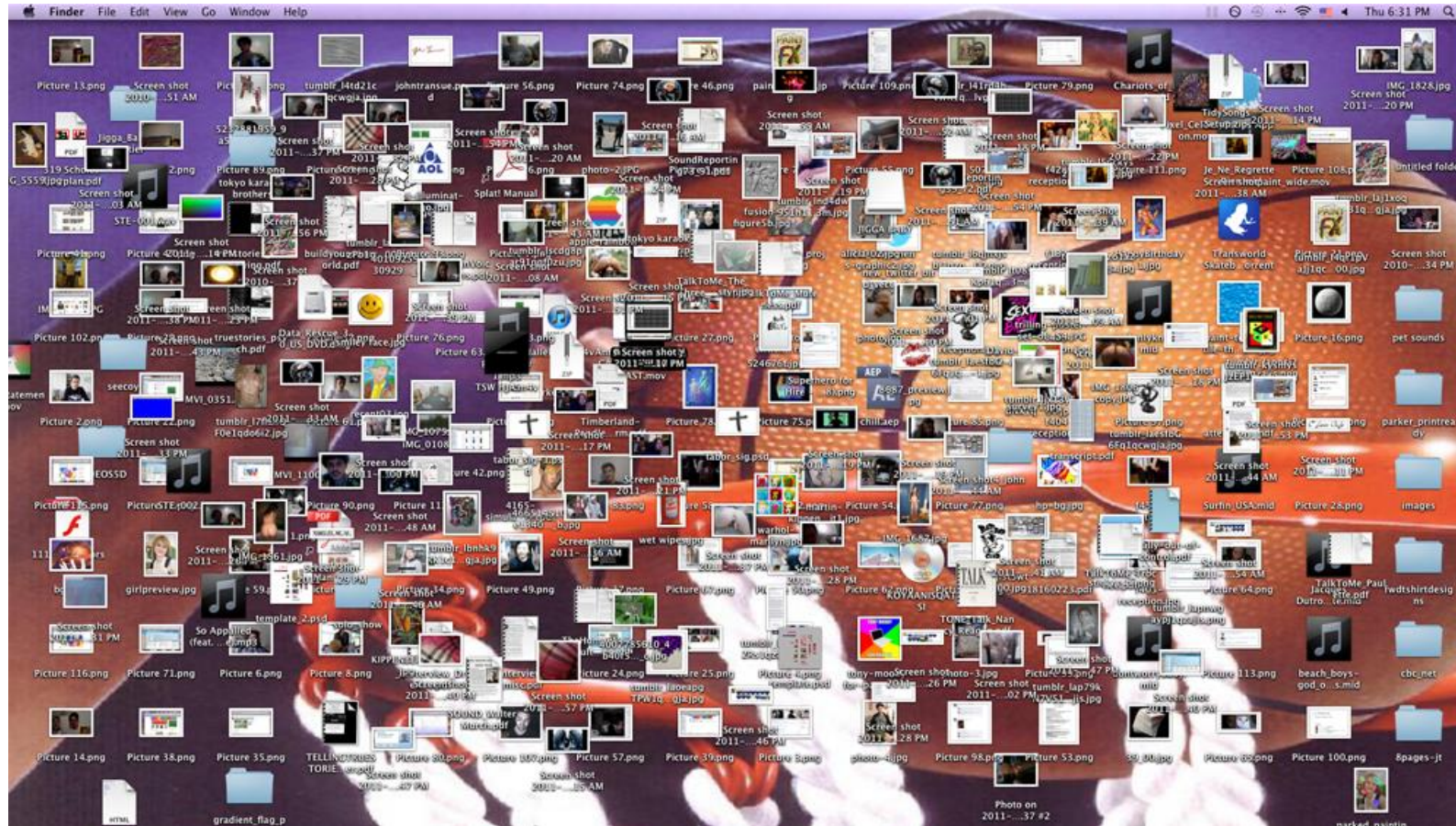
Source: The Turing Way: <https://the-turing-way.netlify.app/>

- Can you reproduce the exact results that you acquired 5 years ago?
- If you cannot reproduce what you have done, how can other people?

|          |           | Data         |               |
|----------|-----------|--------------|---------------|
|          |           | Same         | Different     |
| Analysis | Same      | Reproducible | Replicable    |
|          | Different | Robust       | Generalisable |



# Avoid this at all costs...



## Do not keep things on your desktop!



Reproducibility is your main goal!

A Guide to  
Reproducible Code  
in Ecology and  
Evolution



# Suggestions

Keep all your projects separate!

Use the same project structure:

- Input Data (data)
- Computer code (code/scripts)
- Written documents (doc)
- Calculation output (export/output)

/ > media > adam > work > Dropbox > WorkSpace

| Name                        | Size     | Modified       |
|-----------------------------|----------|----------------|
| 2021-10-12_thermalSelect    | 5 items  | 27.06.22 08:01 |
| 2021-11-20_ordovician-bioge | 1 item   | 20.11.21 11:42 |
| 2021-11-26_habitat          | 7 items  | 19.08.22 14:30 |
| .git                        | 11 items | 15.07.22 22:51 |
| data                        | 4 items  | 07.07.22 11:34 |
| doc                         | 14 items | 08.07.22 16:37 |
| export                      | 17 items | 21.07.22 10:16 |
| scripts                     | 12 items | 21.07.22 15:49 |
| .gitignore                  | 11 B     | 03.12.21 17:11 |
| .projectile                 | 0 B      | 29.11.21 17:24 |
| 2021-12-09_patterson        | 6 items  | 27.05.22 14:47 |
| 2021-12-10_BI               | 1 item   | 10.12.21 08:02 |
| 2022-01-14_datasyntesis     | 1 item   | 14.01.22 18:44 |
| 2022-03-01_bioDeepTime      | 13 items | 04.08.22 15:17 |
| 2022-03-20_parameters       | 3 items  | 20.03.22 18:13 |

2021-10-12\_thermalSelect (folder)

1,3 TiB free



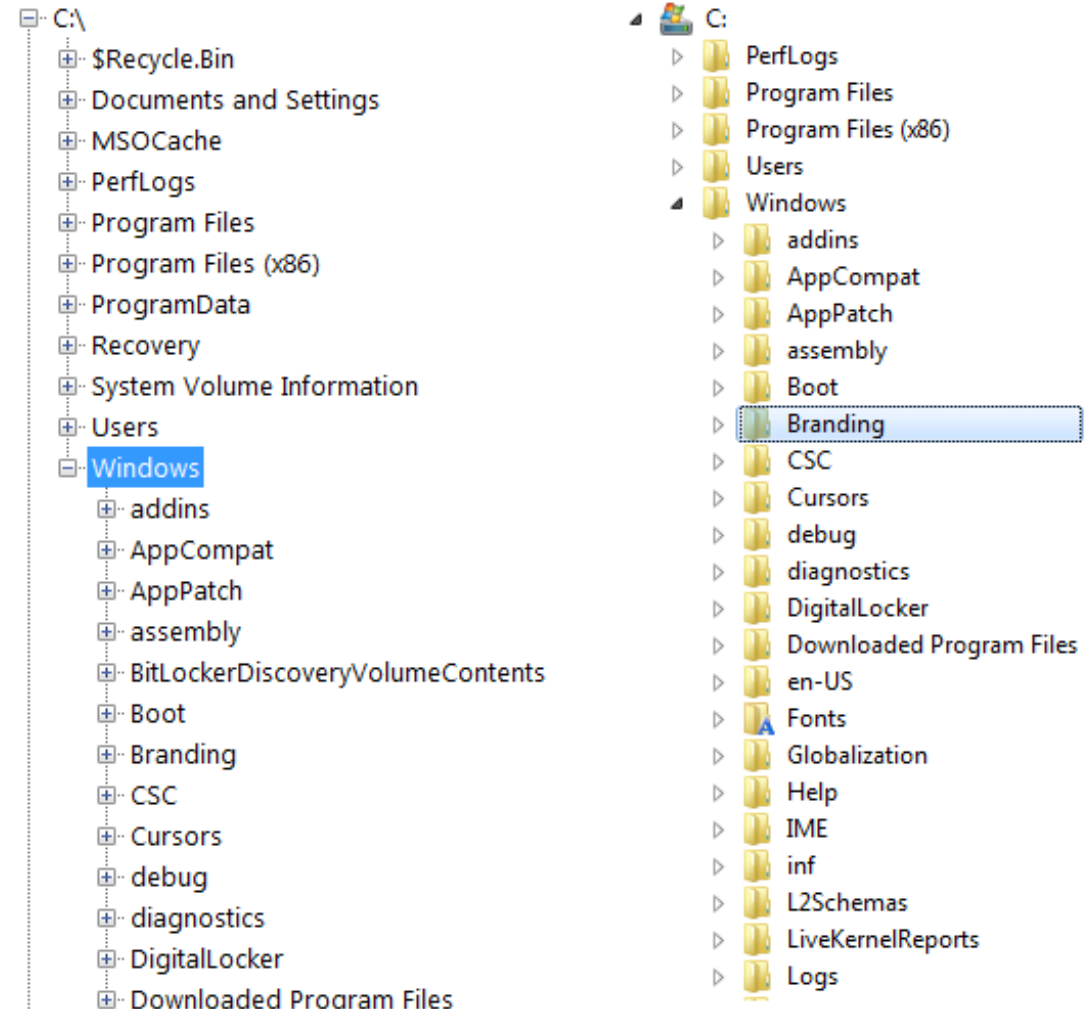
# About files...

# The Windows file system

- Files are data items on storage devices
- Paths use the characteristic backslash \ character to depict nestedness
- Directories are called “Folders”
- File format: filename.ext
- Total path to “Branding”:

C:\Windows\Branding

- Case insensitive!

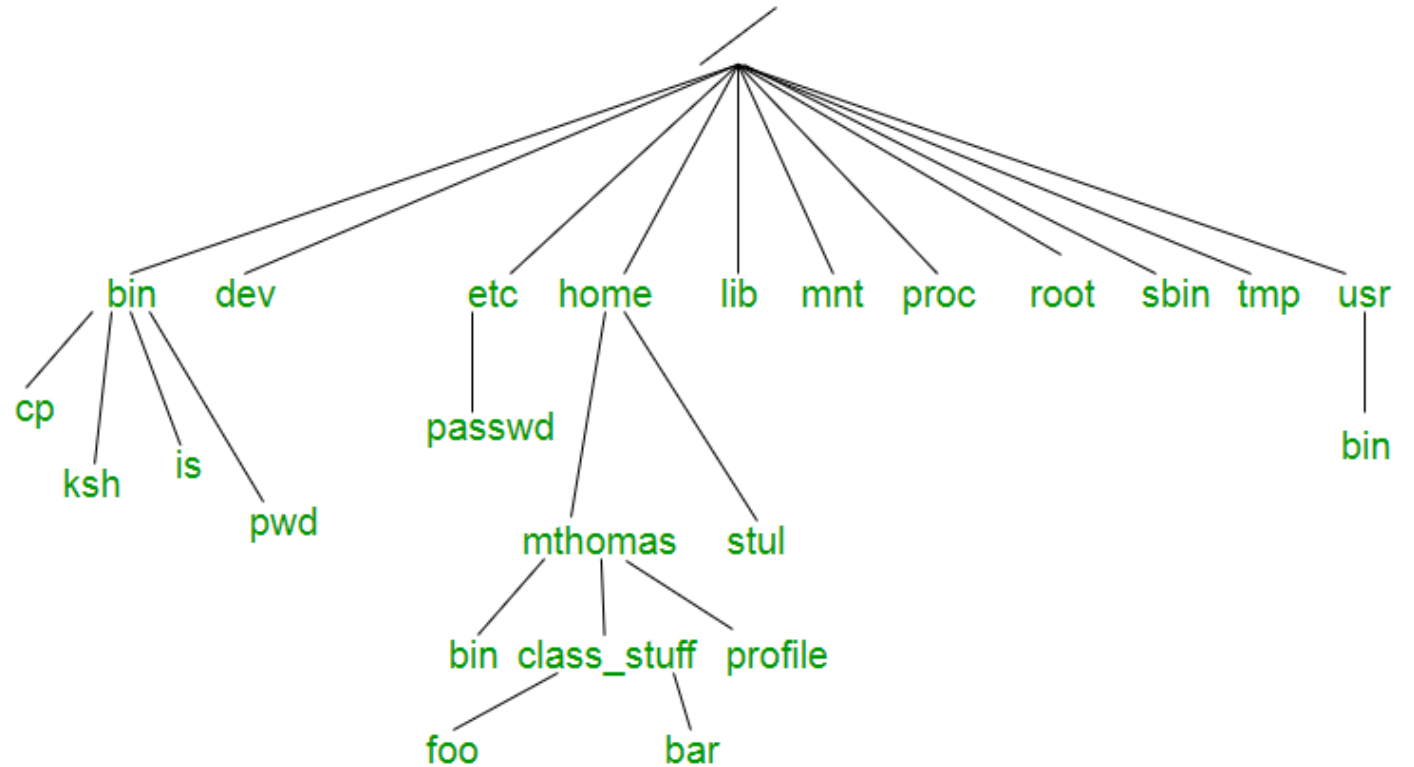


# The UNIX file system

- Shared for UNIX and UNIX-like systems (GNU/Linux, macOS, Android)
- Concept: everything in the computer is represented by a file
- Nestedness coded with forward slash : /
- File format can be anything
- Complete path to “bar”

`/home/mthomas/class_stuff/bar`

- Case sensitive!



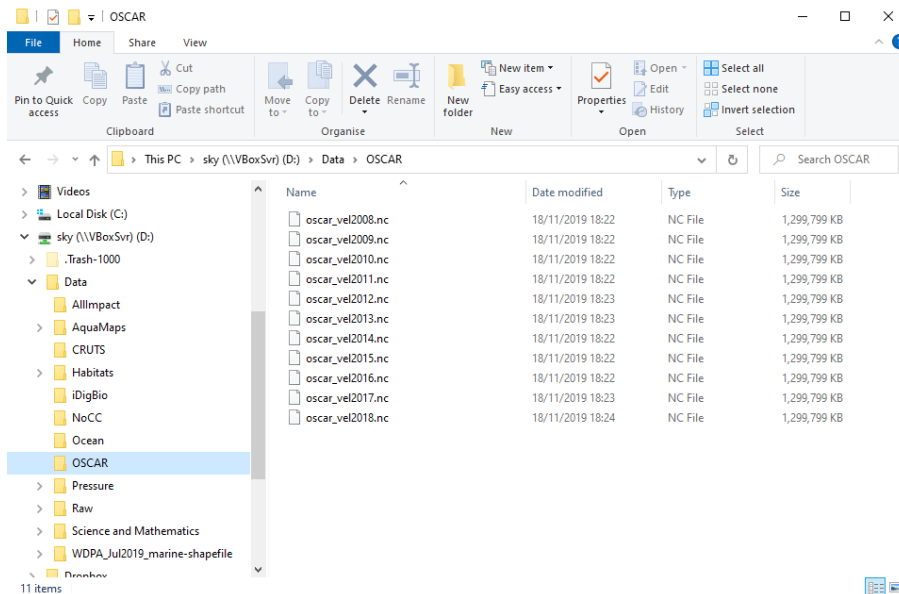
# Two main options:

## Graphical User Interface (GUI)

- Super simple + mouse
- Visually appealing
- “Novice-friendly”

## Command Line Interpreter (CLI)

- Steeper learning curve
- Automation
- Keyboard-only “Expert-friendly”



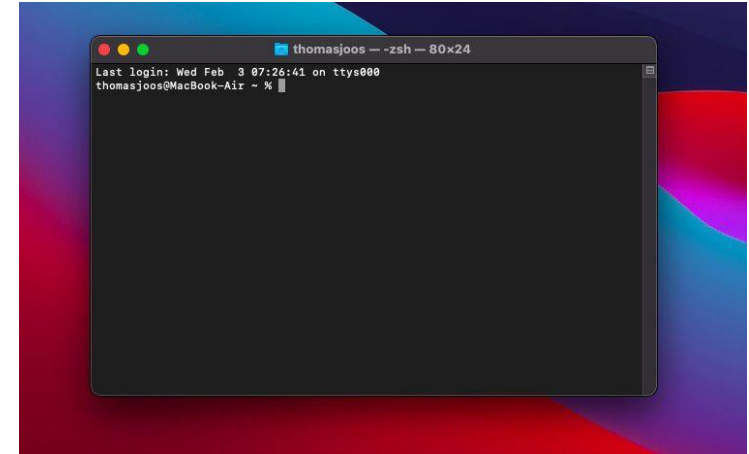
```
adam@posidonia: /mnt/sky/Data/OSCAR 154x19
adam@posidonia:~$ cd /mnt/sky/Data/OSCAR/
adam@posidonia:/mnt/sky/Data/OSCAR$ ls -la
total 14297852
drwxrwxr-x  2 adam adam    4096 Sep  4  2020 .
drwxrwxr-x 14 adam adam    4096 Oct 23  2021 ..
-rwxrwxrwx  1 adam adam 1330993460 Nov 18  2019 oscar_vel2008.nc
-rwxrwxrwx  1 adam adam 1330993460 Nov 18  2019 oscar_vel2009.nc
-rwxrwxrwx  1 adam adam 1330993460 Nov 18  2019 oscar_vel2010.nc
-rwxrwxrwx  1 adam adam 1330993460 Nov 18  2019 oscar_vel2011.nc
-rwxrwxrwx  1 adam adam 1330993512 Nov 18  2019 oscar_vel2012.nc
-rwxrwxrwx  1 adam adam 1330993512 Nov 18  2019 oscar_vel2013.nc
-rwxrwxrwx  1 adam adam 1330993512 Nov 18  2019 oscar_vel2014.nc
-rwxrwxrwx  1 adam adam 1330993512 Nov 18  2019 oscar_vel2015.nc
-rwxrwxrwx  1 adam adam 1330993512 Nov 18  2019 oscar_vel2016.nc
-rwxrwxrwx  1 adam adam 1330993512 Nov 18  2019 oscar_vel2017.nc
-rwxrwxrwx  1 adam adam 1330993512 Nov 18  2019 oscar_vel2018.nc
adam@posidonia:/mnt/sky/Data/OSCAR$
```



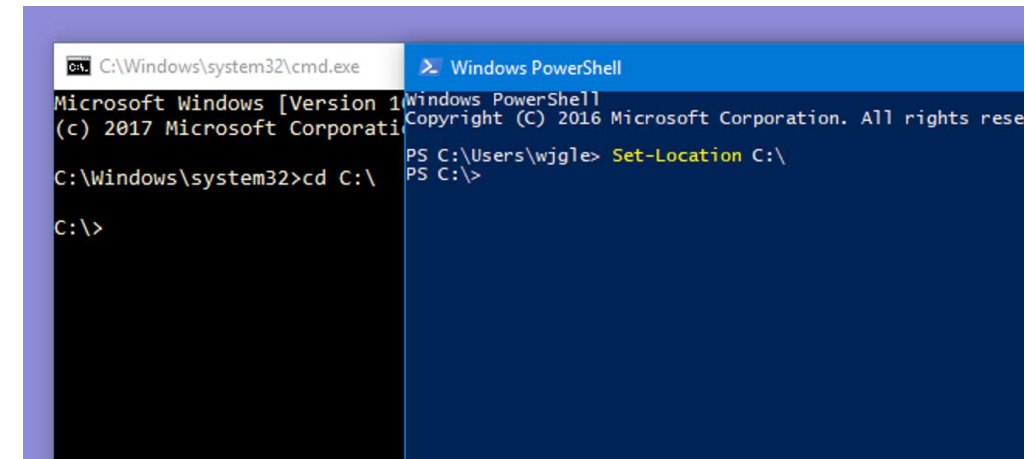
# Terminal emulators

- Every OS has one
- Graphical applications that run a program called the “shell”: an interpreter program that translates instructions
- Console applications can be run with the shell
  - Automation
  - Program building
  - Scientific calculations
- Shells are programmable

## Mac (zsh or bash)

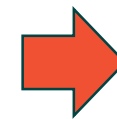


## Windows (cmd and powershell)



# The BASH shell

- Ubiquitous
- Most frequently used on servers and clusters
- UNIX-native: most programming systems use UNIX-like paths – even on Windows!
- Mac: have it, z shell (zsh) is almost the same
- Windows: a simplified version is available with git (git bash)



<https://git-scm.com>

# Installing git for Windows

and GitHub

# BASH essentials

Most important functions and browsing directories



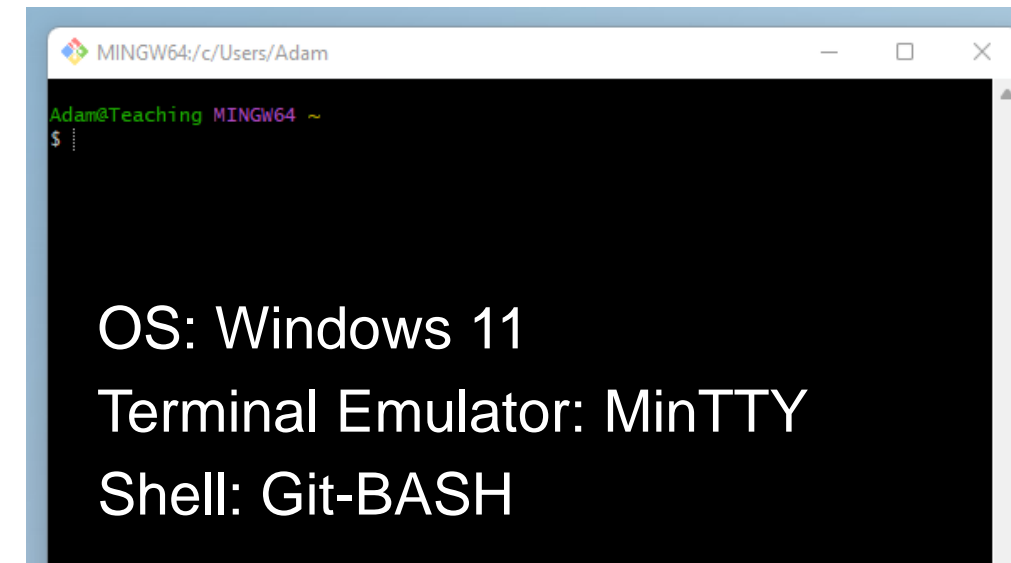
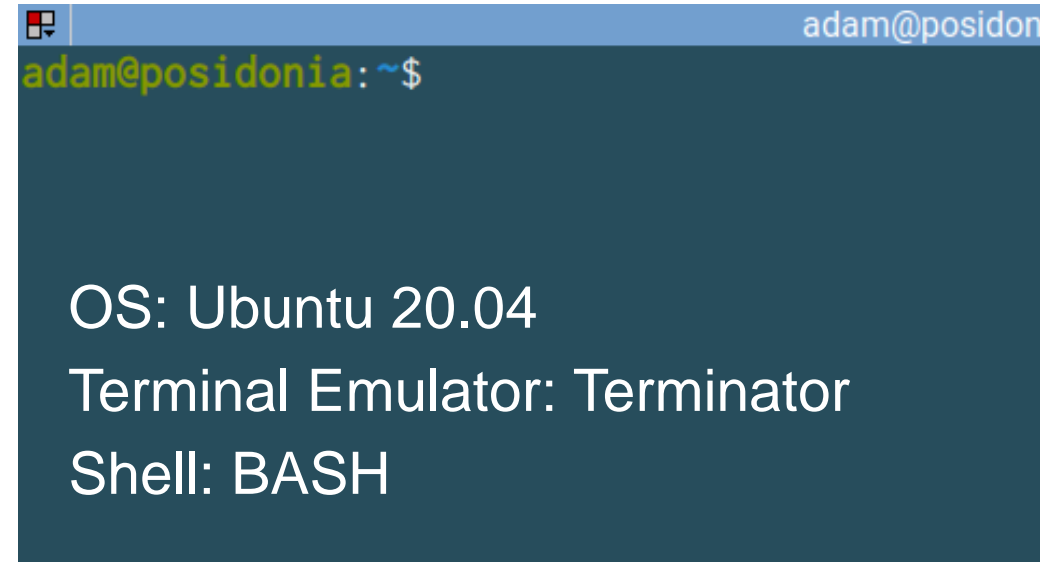
# The prompt

- User input expected (typing)
- Looks different on all, but there are conventions:

`user@host`

`~`: is shorthand for user home

`$`: means normal user mode



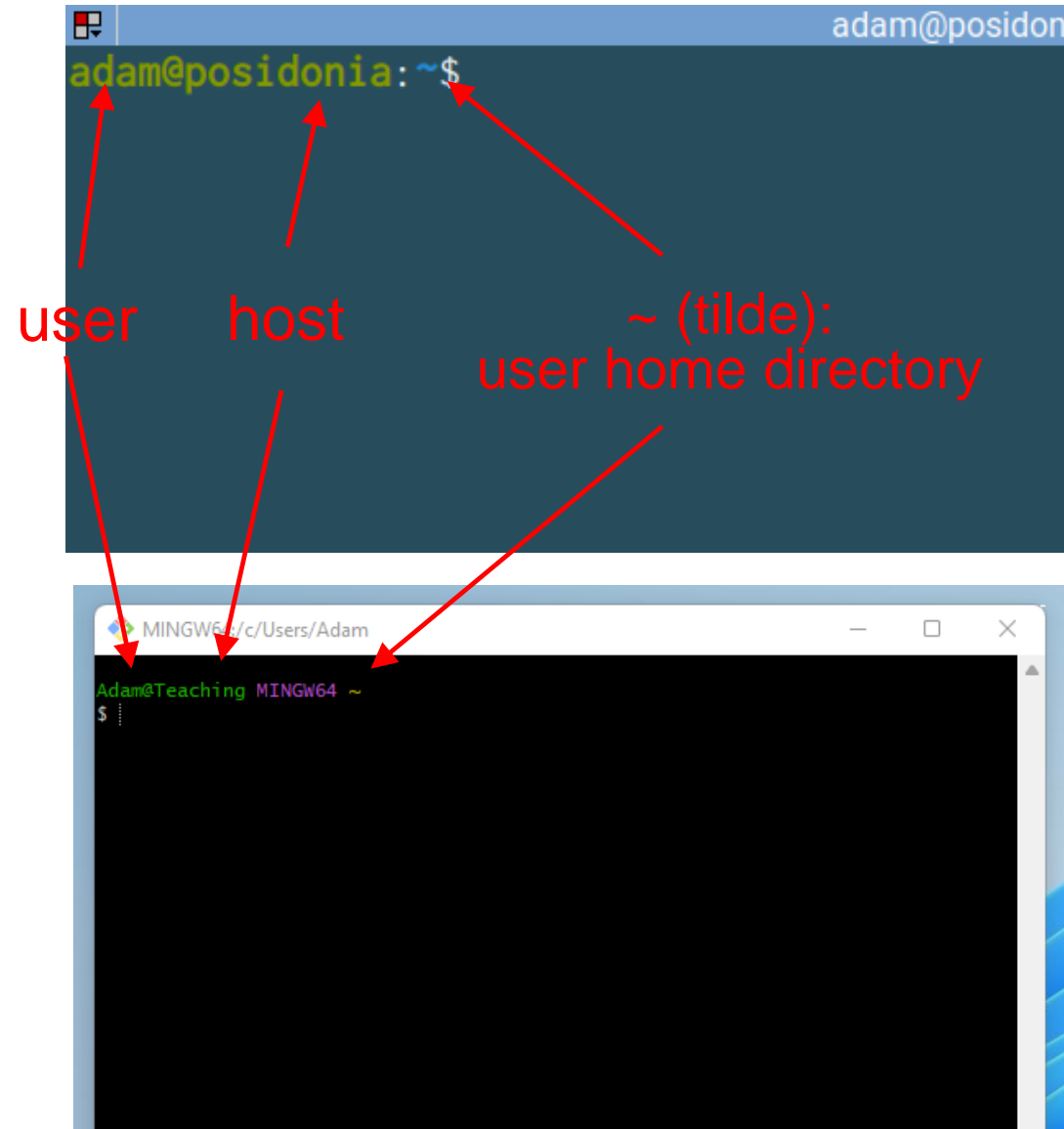
# The prompt

- User input expected (typing)
- Looks different on all, but there are conventions:

`user@host`

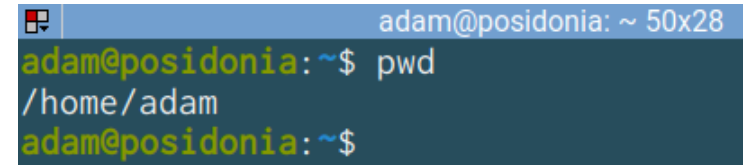
`~`: is shorthand for user home

`$`: means normal user mode



# pwd

**Return path to current directory**



```
adam@posidonia: ~ 50x28
adam@posidonia:~$ pwd
/home/adam
adam@posidonia:~$
```

A terminal window with a dark blue background. The title bar at the top reads "adam@posidonia: ~ 50x28". The terminal shows the user "adam" at host "posidonia" in the home directory "~". The user enters the command "pwd", and the terminal outputs the full path "/home/adam". The prompt returns to "adam@posidonia:~\$".

`mkdir <name>`

space

## Create a directory

- No output to the console: no error occurred (directory was created)

```
adam@posidonia: ~ 50x28  
adam@posidonia:~$ mkdir my_dir  
adam@posidonia:~$
```



# ls

## List directory contents

- Returns a list of entries (both normal files and directories) – can be colored
- Note the quotes around entries with spaces in them!



```
adam@posidonia: ~ 50x28
adam@posidonia:~$ ls
01-network-manager-all.yaml  Programs
1_linktags.sh                Public
Desktop                      random.conf
Documents                    report
Downloads                    snap
Edraw                        some.df
exercises                    Templates
gems                         temp.mbsync
Mail                         Videos
Music                       virtual
my_dir                       'VirtualBox VMs'
'NVIDIA Nsight Systems'     zen.json
Pictures                    Zotero
adam@posidonia:~$
```

ls -l

## List directory contents (with option l)

- Long output, includes attributes

d:directory

permissions

owner

size (bytes)

modification

name

```
adam@posidonia: ~ 73x30
adam@posidonia:~$ ls -l
total 112
-rw-r--r-- 1 adam adam 104 Sep  2  2020 01-network-manager-all.yaml
-rw-r--r-- 1 adam adam 224 Jul  2 13:34 1_linktags.sh
drwxr-xr-x 3 adam adam 4096 Jul 29 19:58 Desktop
drwxrwxr-x 2 adam adam 4096 Dez 16  2021 Documents
drwxr-xr-x 5 adam adam 16384 Aug 19 12:00 Downloads
drwxrwxr-x 3 adam adam 4096 Sep  5  2020 Edraw
drwxrwxr-x 3 adam adam 4096 Okt 14  2021 exercises
drwxrwxr-x 10 adam adam 4096 Aug 15 14:40 gems
drwxrwxr-x 7 adam adam 4096 Jan 24  2022 Mail
drwxr-xr-x 2 adam adam 4096 Sep  1  2020 Music
drwxrwxr-x 2 adam adam 4096 Aug 19 16:09 my_dir
drwxrwxr-x 2 adam adam 4096 Apr 21 15:34 'NVIDIA Nsight Systems'
drwxr-xr-x 2 adam adam 4096 Sep  1  2020 Pictures
drwxrwxr-x 4 adam adam 4096 Apr 28  2021 Programs
drwxr-xr-x 2 adam adam 4096 Sep  1  2020 Public
-rw-rw-r-- 1 adam adam  9 Aug 17 17:39 random.conf
drwxrwxr-x 3 adam adam 4096 Jun  2 12:36 report
drwx----- 5 adam adam 4096 Jun  4  2021 snap
drwxr-xr-x 2 adam adam 4096 Okt 23  2020 some.df
drwxr-xr-x 2 adam adam 4096 Sep  1  2020 Templates
-rw-rw-r-- 1 adam adam  0 Jan 24  2022 temp.mbsync
drwxr-xr-x 3 adam adam 4096 Mai 16 16:07 Videos
drwxrwxr-x 3 adam adam 4096 Jan 25  2019 virtual
drwxrwxr-x 5 adam adam 4096 Mai 30 10:38 'VirtualBox VMs'
-rw-rw-r-- 1 adam adam 154 Mär 16 13:11 zen.json
drwxr-xr-x 9 adam adam 4096 Aug 18 22:36 Zotero
adam@posidonia:~$
```

# cd\_<path\_to\_directory>

## Go to a directory

- Can be relative or absolute!

```
adam@posidonia: ~/my_dir 73x30
adam@posidonia:~$ cd my_dir
adam@posidonia:~/my_dir$
```

```
adam@posidonia: ~/my_dir 73x30
adam@posidonia:~$ cd /home/adam/my_dir
adam@posidonia:~/my_dir$
```

`cd _ . .`

## Go to parent directory

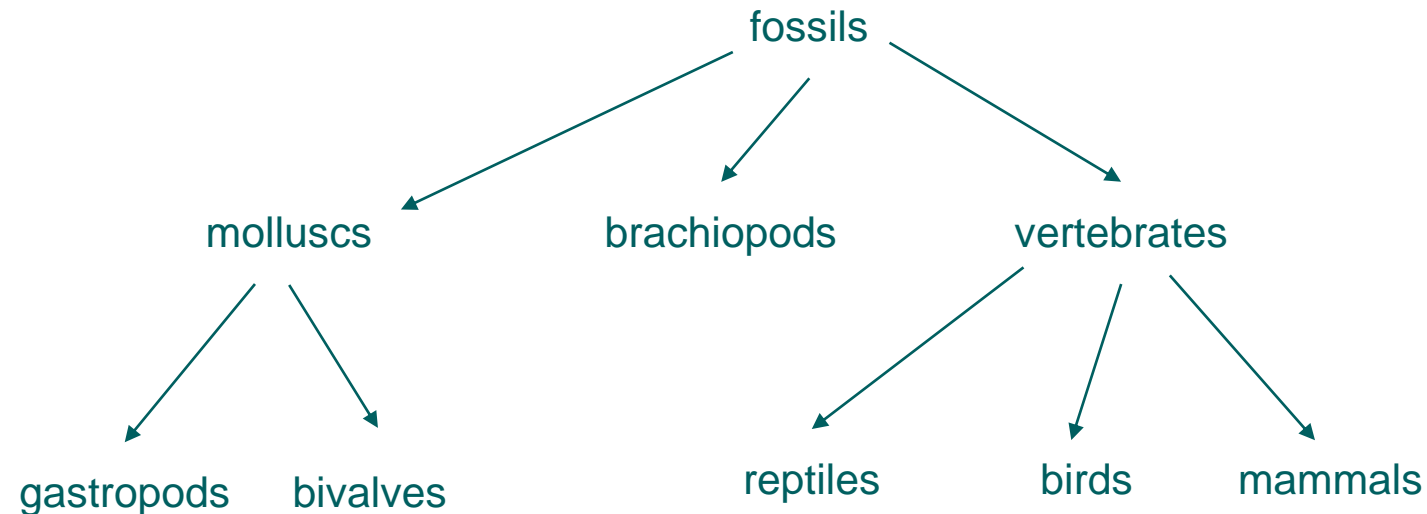
- .. (dot dot) is a placeholder for the parent of the current directory (one up in the hierarchy)

```
adam@posidonia: ~ 73x30
adam@posidonia:~$ cd /home/adam/my_dir
adam@posidonia:~/my_dir$ cd ..
adam@posidonia:~$
```



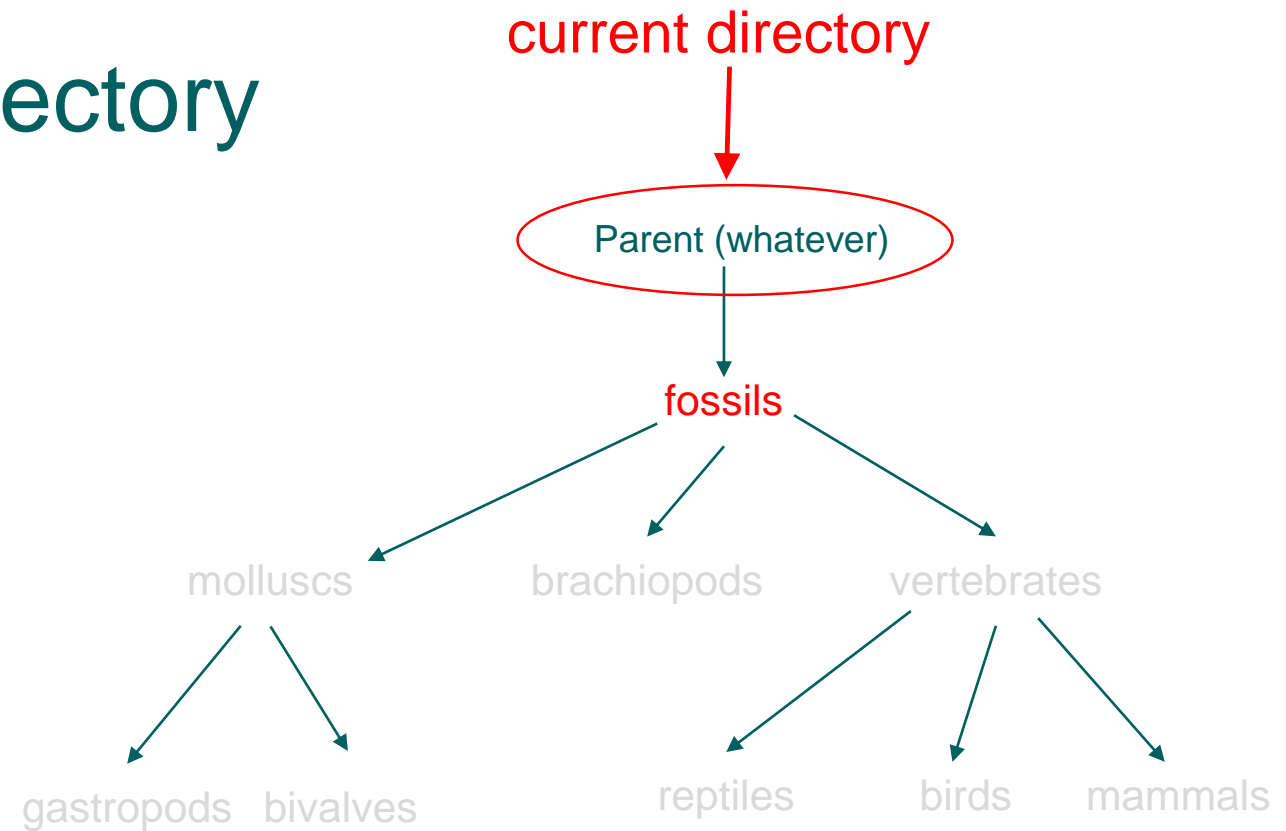
# Exercise (5 minutes)

- Create this directory structure using the combinations of the previous commands!



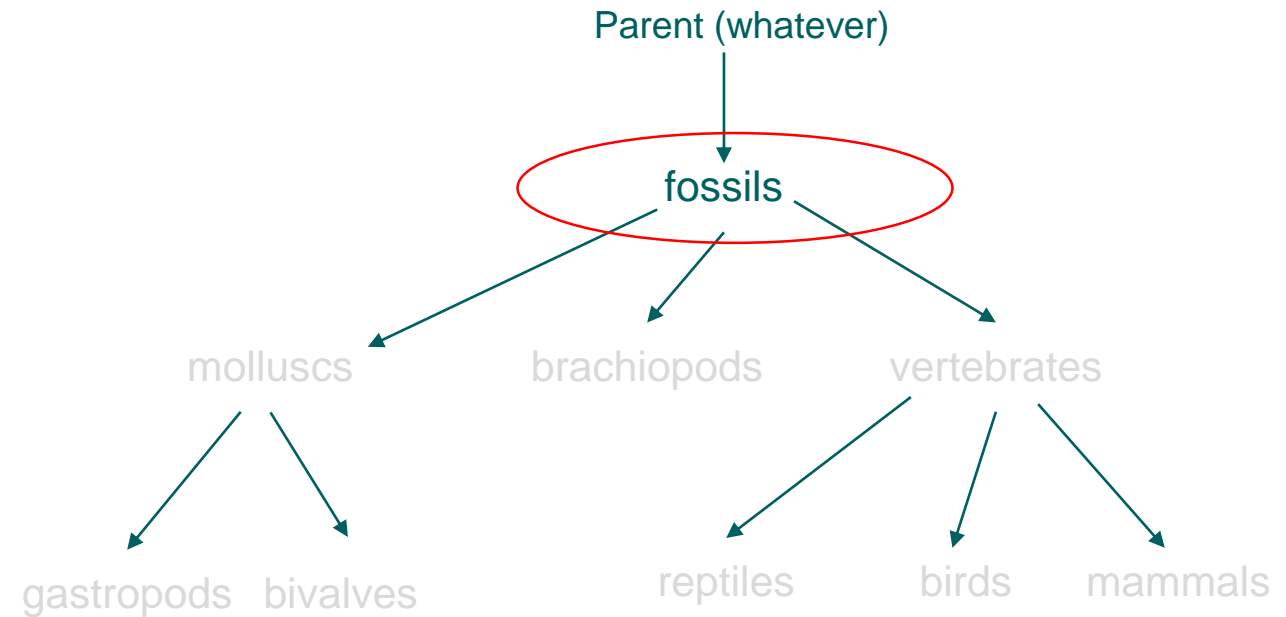
# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```



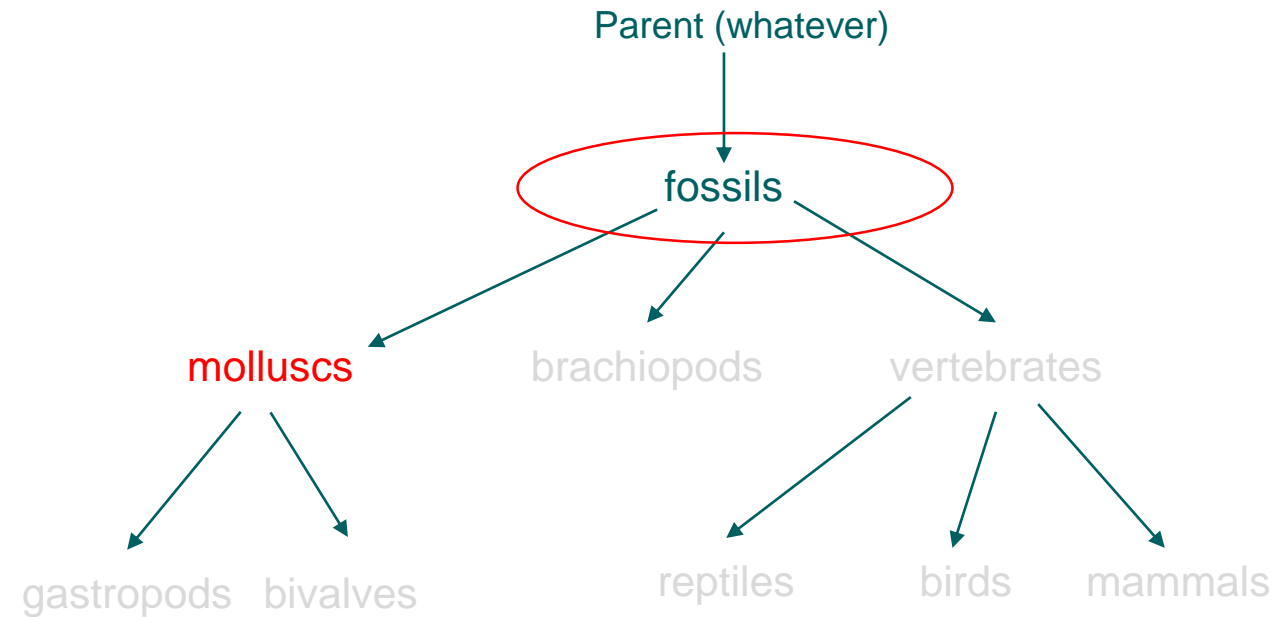
# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```



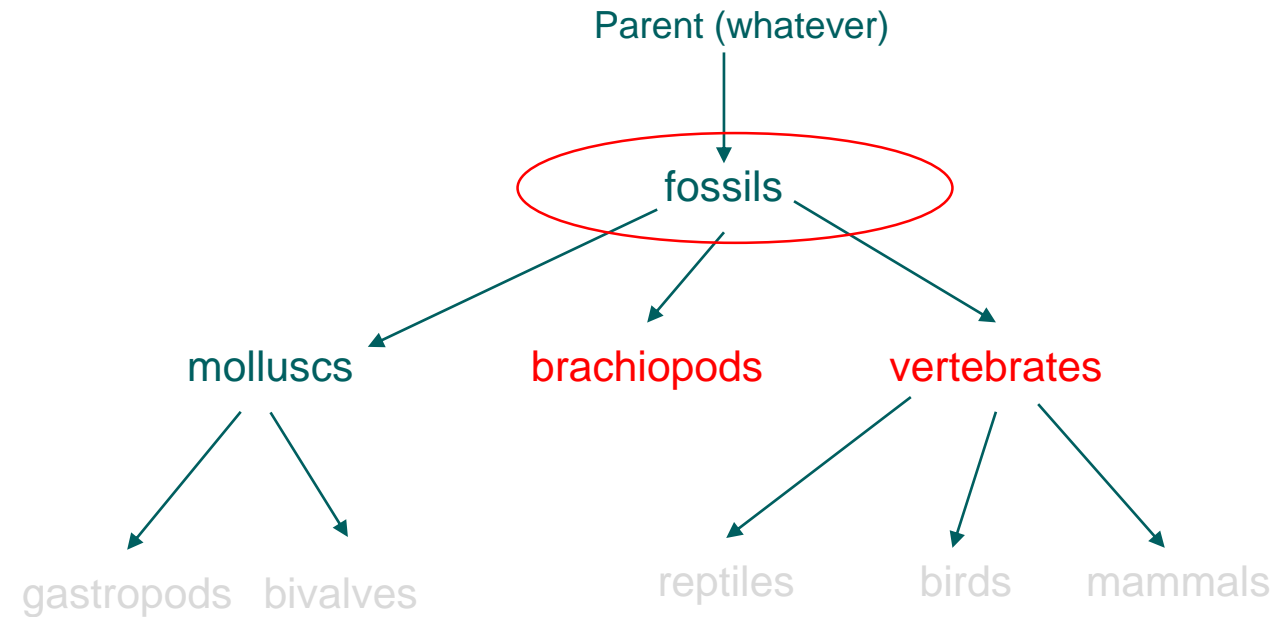
# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```



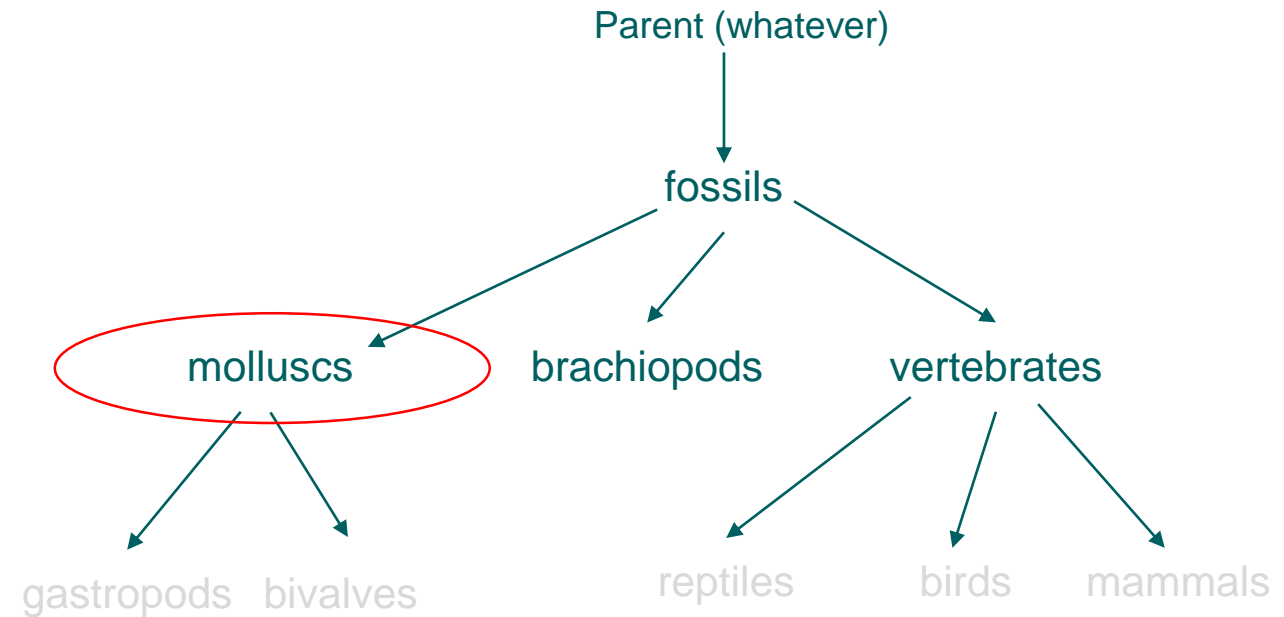
# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```



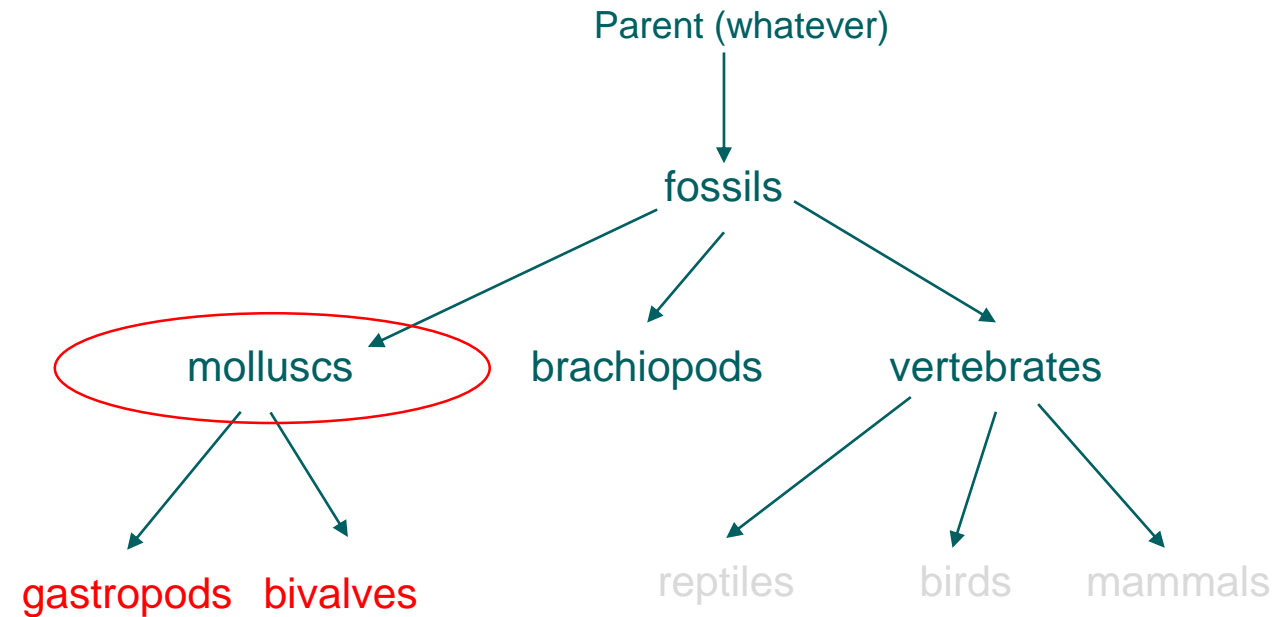
# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```



# Solution 1 – changing directory

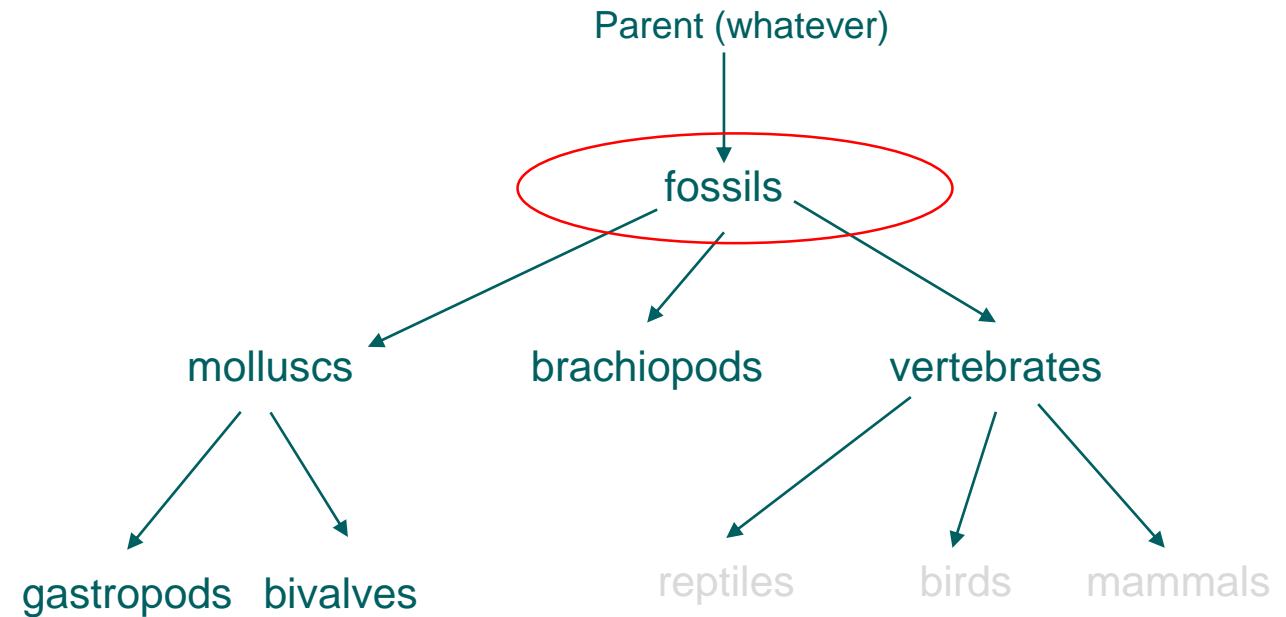
```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```





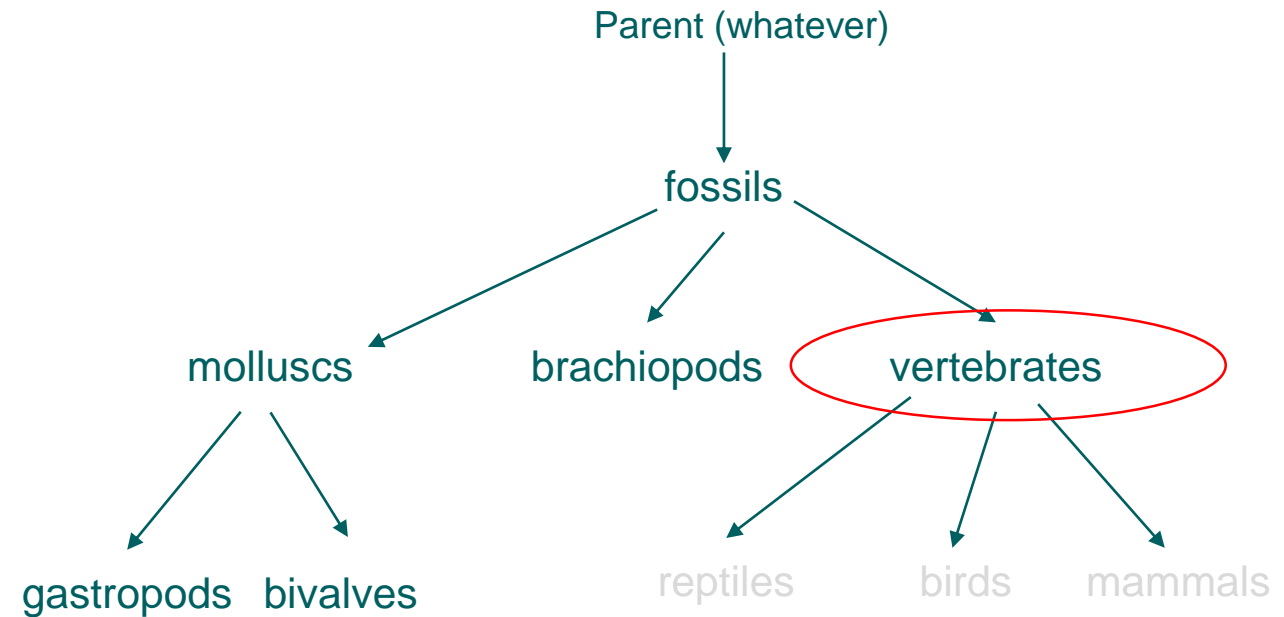
# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```



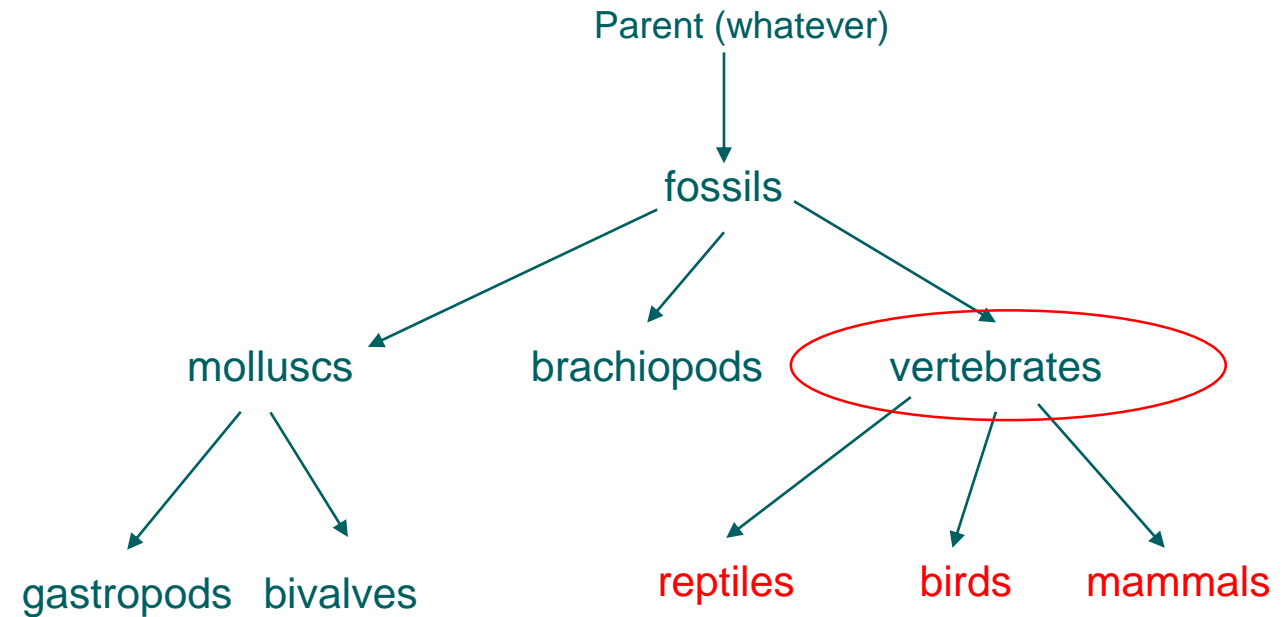
# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```



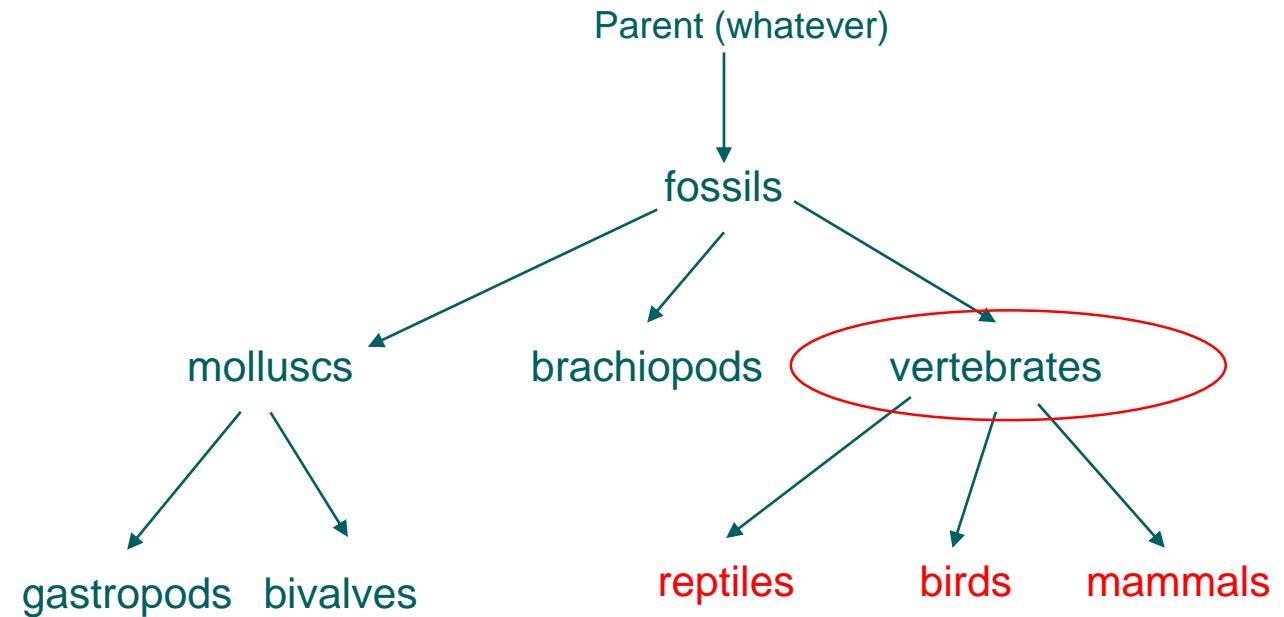
# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```



# Solution 1 – changing directory

```
adam@posidonia: ~/fossils/vertebrates 73x30
adam@posidonia:~$ mkdir fossils
adam@posidonia:~$ cd fossils
adam@posidonia:~/fossils$ mkdir molluscs
adam@posidonia:~/fossils$ mkdir brachiopods vertebrates
adam@posidonia:~/fossils$ cd molluscs
adam@posidonia:~/fossils/molluscs$ mkdir gastropods bivalves
adam@posidonia:~/fossils/molluscs$ cd ..
adam@posidonia:~/fossils$ cd vertebrates
adam@posidonia:~/fossils/vertebrates$ mkdir reptiles birds mammals
adam@posidonia:~/fossils/vertebrates$
```

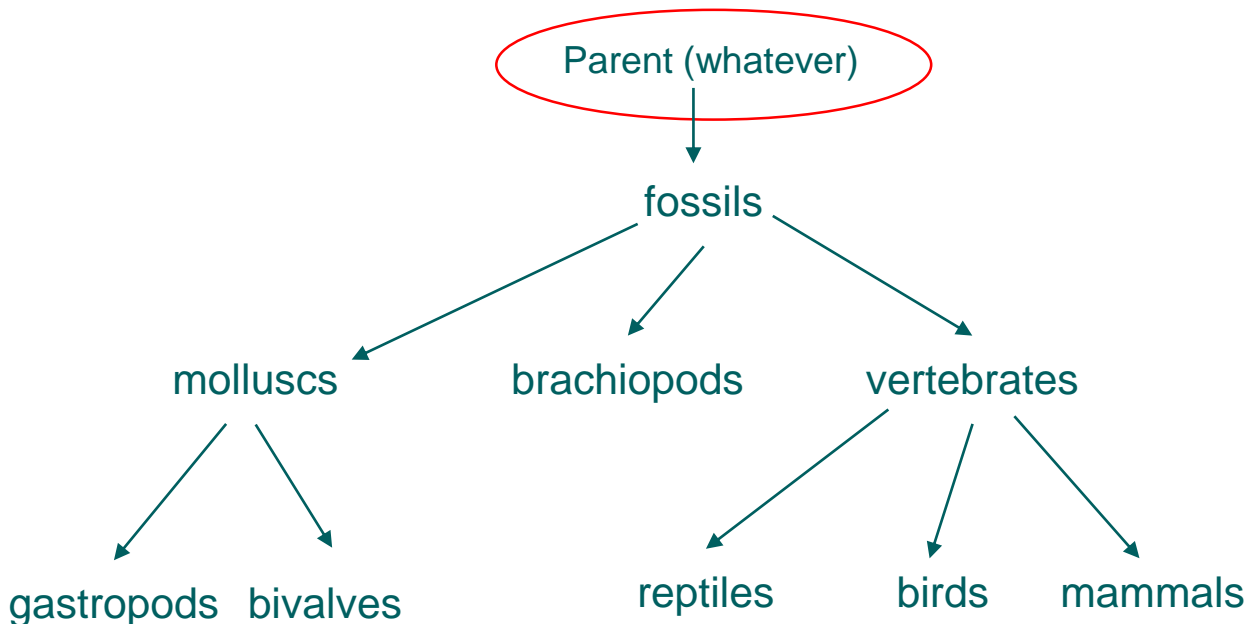


**How to check?**

# find\_<path to directory>

## Recursive listing

- 1. Go back to the parent
- 2. Use find there!



```
adam@posidonia: ~ 73x30
adam@posidonia:~/fossils/vertebrates$ cd ../../
adam@posidonia:~$ find fossils
fossils
fossils/molluscs
fossils/molluscs/gastropos
fossils/molluscs/bivalves
fossils/brachiopods
fossils/vertebrates
fossils/vertebrates/reptiles
fossils/vertebrates/birds
fossils/vertebrates/mammals
adam@posidonia:~$
```

parent of parent

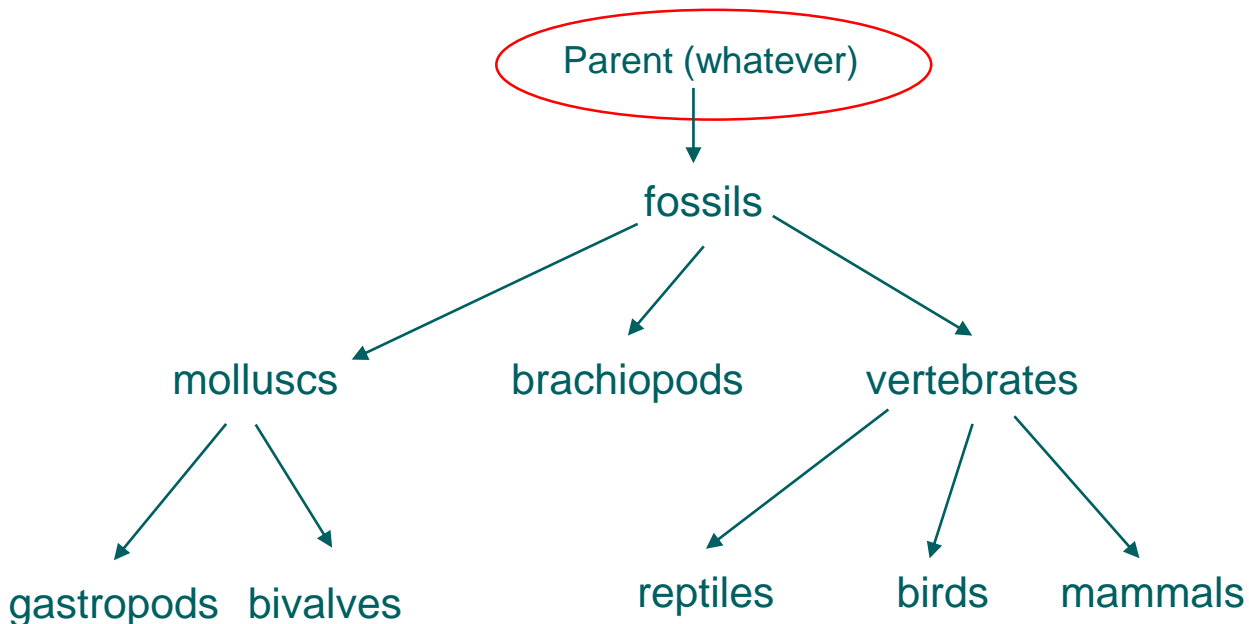
relative paths

`find` `<path>` `>` `<path_to file>`

Angled bracket or ***chevron***

## Output redirection

- Whatever was output to the console is now in a new file!



```
adam@posidonia: ~ 73x30
adam@posidonia:~/fossils/vertebrates$ cd ../../
adam@posidonia:~$ find fossils
fossils
fossils/molluscs
fossils/molluscs/gastropos
fossils/molluscs/bivalves
fossils/brachiopods
fossils/vertebrates
fossils/vertebrates/reptiles
fossils/vertebrates/birds
fossils/vertebrates/mammals
adam@posidonia:~$ find fossils > fossil_path.txt
adam@posidonia:~$
```

***">" Will overwrite existing files!***

# Suggested nomenclature

( ) : **Parenthesis** (open and close)  
[ ] : **Bracket** (open and close)  
{ } : **Brace** (open and close)  
< > : **Chevrons** (left and right)

# cat\_<path to file>

## Display contents of file

- Exactly as it was output to the console

```
adam@posidonia: ~ 73x30
adam@posidonia:~$ cat fossil_path.txt
fossils
fossils/molluscs
fossils/molluscs/gastropods
fossils/molluscs/bivalves
fossils/brachiopods
fossils/vertebrates
fossils/vertebrates/reptiles
fossils/vertebrates/birds
fossils/vertebrates/mammals
adam@posidonia:~$
```



`rm -r <path to dir>`

### Recursive deletion (-r)

- Deletes the content of the directory and the directory itself
- `rmdir` doesn't work! for the
- No output = success?!

```
adam@posidonia: ~ 73x30
adam@posidonia:~$ cat fossil_path.txt
fossils
fossils/molluscs
fossils/molluscs/gastropods
fossils/molluscs/bivalves
fossils/brachiopods
fossils/vertebrates
fossils/vertebrates/reptiles
fossils/vertebrates/birds
fossils/vertebrates/mammals
adam@posidonia:~$ rm -r fossils ←
adam@posidonia:~$
```

**WARNING!**  
The results of `rm`  
cannot be undone!

# echo <text>

## Print something

- Used to print things to the console (standard output)
- \$? Is a special symbol: the exit code of the last command:
  - 0: Success
  - Other: Failure

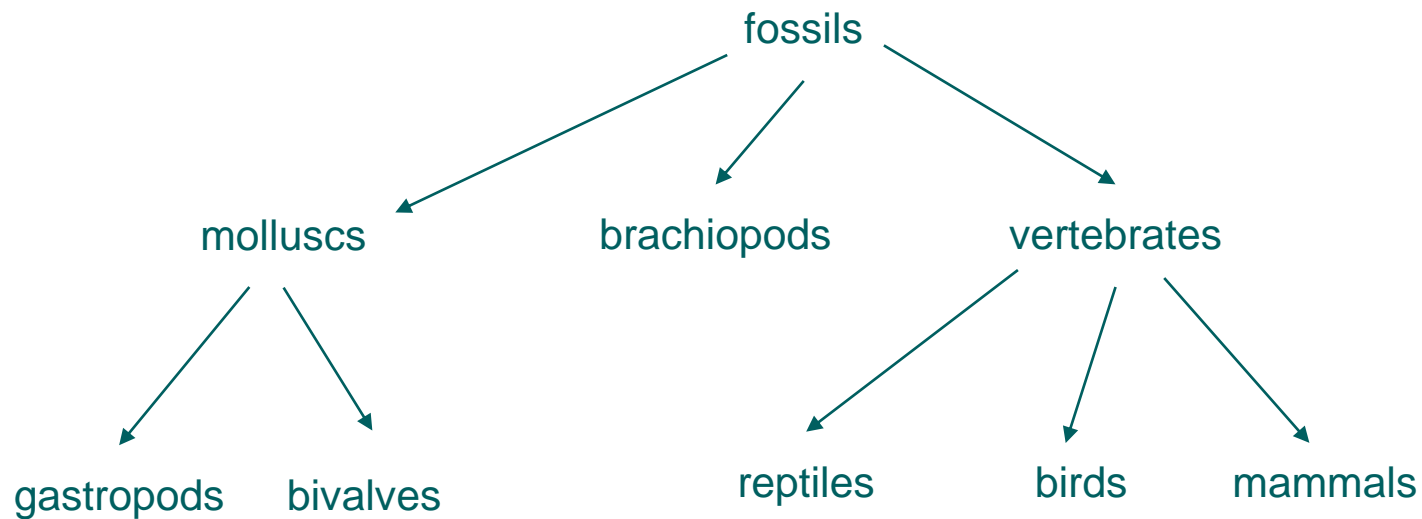
<https://www.redhat.com/sysadmin/exit-codes-demystified>

adam@posidonia: ~ 73x30

```
adam@posidonia:~$ cat fossil_path.txt
fossils
fossils/molluscs
fossils/molluscs/gastropods
fossils/molluscs/bivalves
fossils/brachiopods
fossils/vertebrates
fossils/vertebrates/reptiles
fossils/vertebrates/birds
fossils/vertebrates/mammals
adam@posidonia:~$ rm -r fossils
adam@posidonia:~$ echo $? ←
0
adam@posidonia:~$
```

# Recreate the structure!

- Did you type things into the console?!



Hint 1. Use a general-purpose text editor!

Novice-friendly:

- Sublime Text



- VS Code



- Atom



Expert-friendly:

- Vim

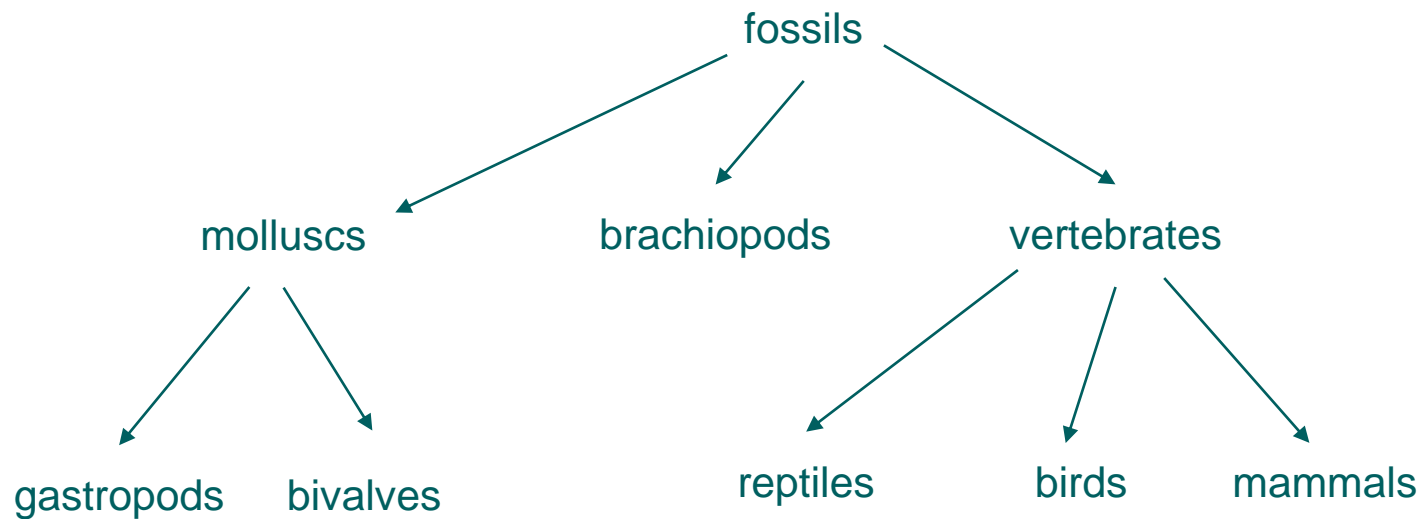


- Emacs



# Solution 2

- Using the same reference directory



**Hint 2. we can use the contents of fossil\_path.txt**

Add mkdir in front of every line, then copy and paste into the console!

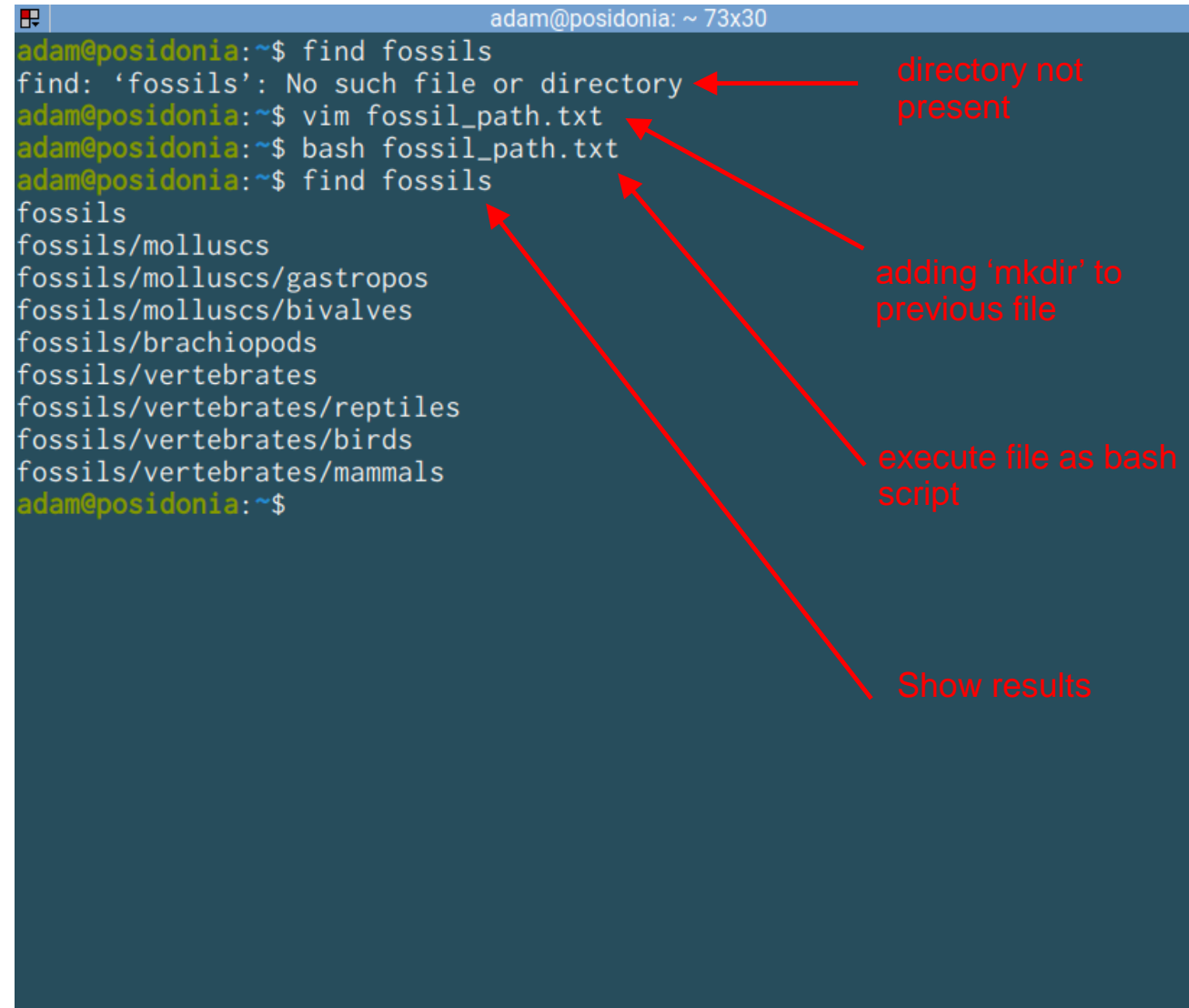
```
1 mkdir fossils
2 mkdir fossils/molluscs
3 mkdir fossils/molluscs/gastropods
4 mkdir fossils/molluscs/bivalves
5 mkdir fossils/brachiopods
6 mkdir fossils/vertebrates
7 mkdir fossils/vertebrates/reptiles
8 mkdir fossils/vertebrates/birds
9 mkdir fossils/vertebrates/mammals
```

`bash_<path>`

## Executing shell scripts

- The text we created is actually a shell script
- The “bash” console application program can be used to execute it.

<https://www.redhat.com/sysadmin/exit-codes-demystified>



The terminal window shows the following sequence of commands and output:

```
adam@posidonia: ~ 73x30
adam@posidonia:~$ find fossils
find: 'fossils': No such file or directory
adam@posidonia:~$ vim fossil_path.txt
adam@posidonia:~$ bash fossil_path.txt
adam@posidonia:~$ find fossils
fossils
fossils/molluscs
fossils/molluscs/gastropods
fossils/molluscs/bivalves
fossils/brachiopods
fossils/vertebrates
fossils/vertebrates/reptiles
fossils/vertebrates/birds
fossils/vertebrates/mammals
adam@posidonia:~$
```

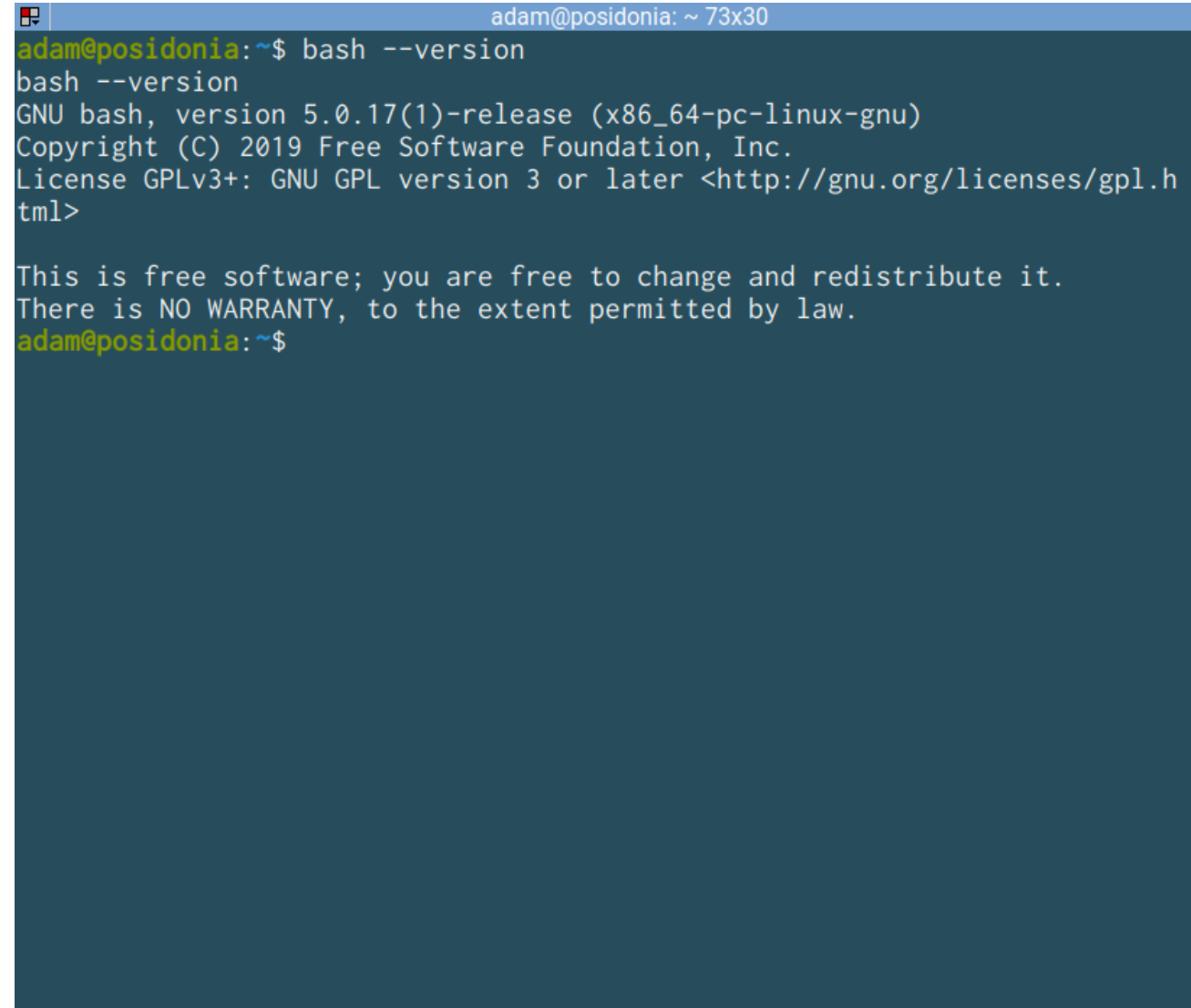
Annotations with red arrows point to specific parts of the terminal output:

- directory not present**: Points to the error message "find: 'fossils': No such file or directory".
- adding 'mkdir' to previous file**: Points to the command "vim fossil\_path.txt".
- execute file as bash script**: Points to the command "bash fossil\_path.txt".
- Show results**: Points to the output of the "find fossils" command.

# bash\_--version

## Running console applications

- `--version`: ask for program version
- `--help`: display help for program

A terminal window with a blue title bar containing the text 'adam@posidonia: ~ 73x30'. The terminal has a dark blue background with white text. The command 'bash --version' has been entered and executed. The output shows the version of GNU bash (5.0.17(1)-release) and its license (GPLv3+). It also includes a disclaimer: 'This is free software; you are free to change and redistribute it. There is NO WARRANTY, to the extent permitted by law.' The prompt 'adam@posidonia:~\$' is visible at the bottom.

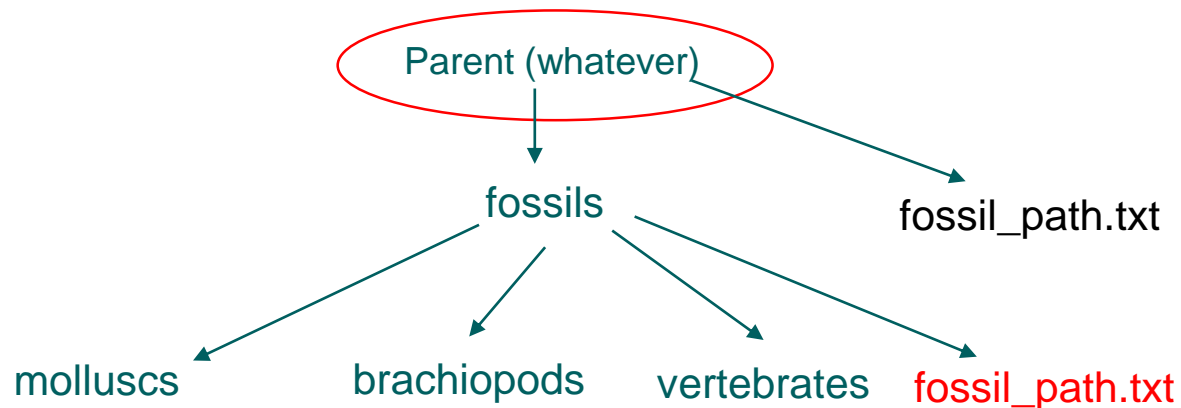
```
adam@posidonia:~$ bash --version
bash --version
GNU bash, version 5.0.17(1)-release (x86_64-pc-linux-gnu)
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
adam@posidonia:~$
```

# cp\_<what>\_<where>

## Copying a file or directory

- Target directory or file
- If directory, the file will be put into it



```
adam@posidonia: ~ 73x30
adam@posidonia:~$ cp fossil_path.txt fossils
adam@posidonia:~$ ls fossils
brachiopods fossil_path.txt molluscs vertebrates
adam@posidonia:~$
```

List contents of path!

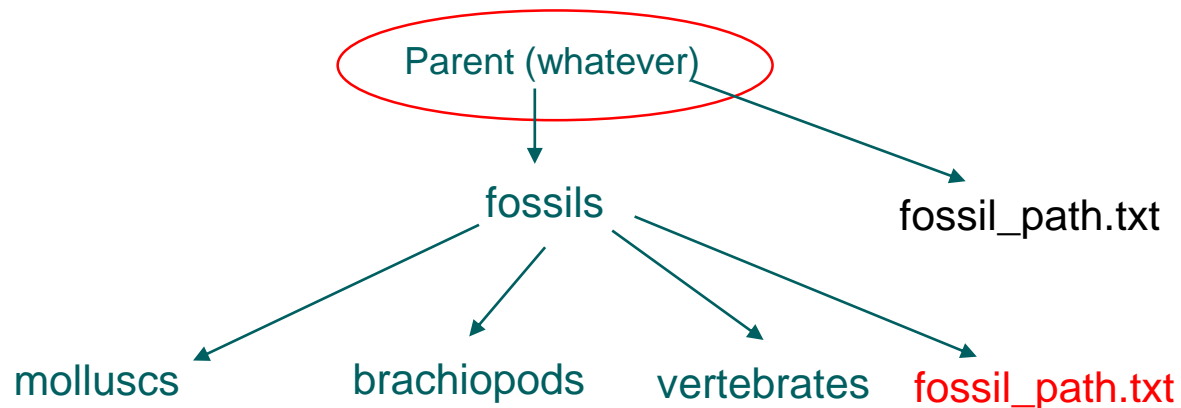
New file

One of the tools that we looked at can be used to delete the file that we have created. Try to delete it!

# rm\_<path\_file>

**Without -r removes a single file**

- As with cp, multiple files can be passed to this (separated by spaces)



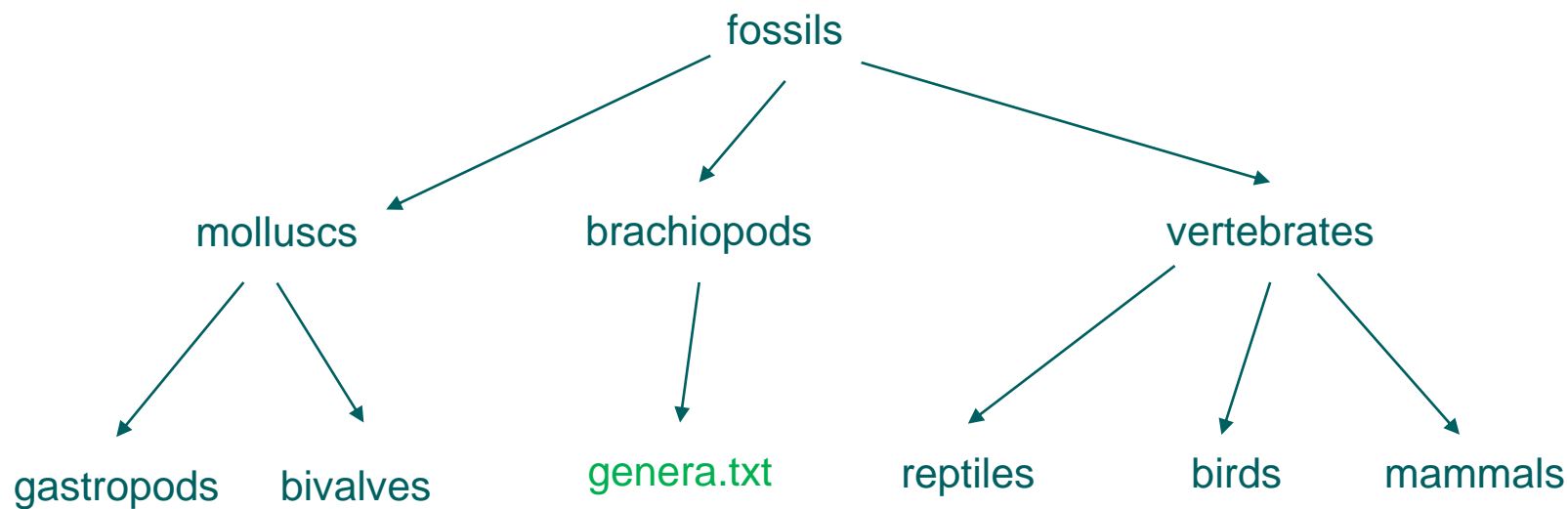
```
adam@posidonia: ~ 73x30
adam@posidonia:~$ cp fossil_path.txt fossils
adam@posidonia:~$ ls fossils
brachiopods fossil_path.txt molluscs vertebrates
adam@posidonia:~$ rm fossils/fossil_path.txt
adam@posidonia:~$ ls fossils
brachiopods molluscs vertebrates
adam@posidonia:~$
```

File disappeared



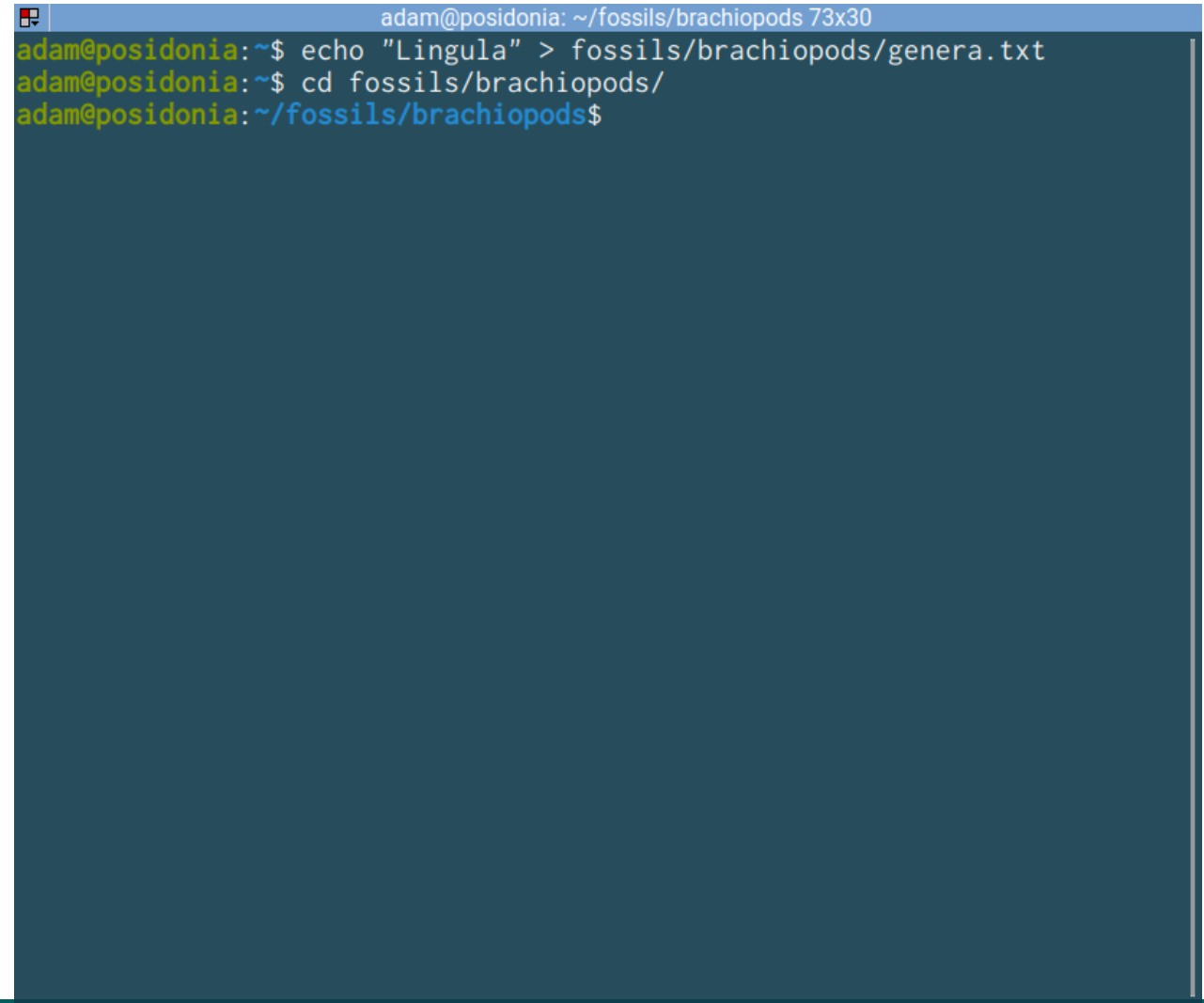
# Exercise!

1. Use an `echo` statement to write the genus name “*Lingula*” into `fossils/brachiopods/genera.txt`!
2. Then change directory to `brachiopods`.



```
echo "Lingula" > fossils/brachiopods/genera.txt
```

- You can use the double chevron  
>> to append to an existing file

A terminal window with a dark blue background and light blue title bar. The title bar text is 'adam@posidonia: ~/fossils/brachiopods 73x30'. The terminal shows three lines of text: 'adam@posidonia:~\$ echo "Lingula" > fossils/brachiopods/genera.txt', 'adam@posidonia:~\$ cd fossils/brachiopods/', and 'adam@posidonia:~/fossils/brachiopods\$'.

```
adam@posidonia: ~/fossils/brachiopods 73x30
adam@posidonia:~$ echo "Lingula" > fossils/brachiopods/genera.txt
adam@posidonia:~$ cd fossils/brachiopods/
adam@posidonia:~/fossils/brachiopods$
```

# echo "Spiriferina" >> genera.txt

## Appending to files

- You can use the double chevron >> to append to an existing file
- Added to new line!

```
adam@posidonia: ~/fossils/brachiopods 73x30
adam@posidonia:~$ echo "Lingula" > fossils/brachiopods/genera.txt
adam@posidonia:~$ cd fossils/brachiopods/
adam@posidonia:~/fossils/brachiopods$ echo "Spiriferina" >> genera.txt
adam@posidonia:~/fossils/brachiopods$ cat genera.txt
Lingula
Spiriferina
adam@posidonia:~/fossils/brachiopods$
```

# Special characters

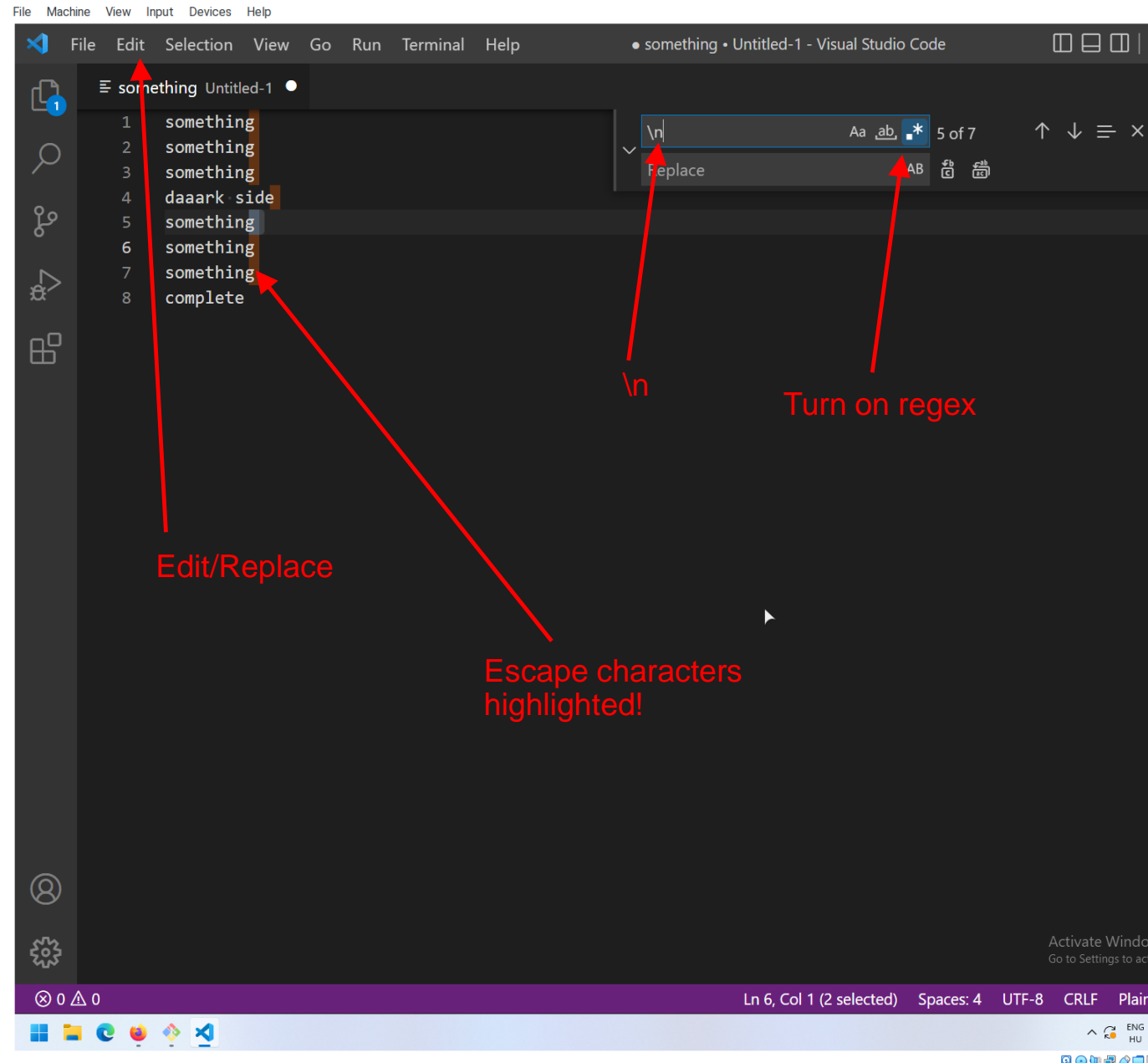
We use so called “escape characters to denote special symbols, that sometimes have other meanings.

`\n`: newline escape

`\t`: tab escape

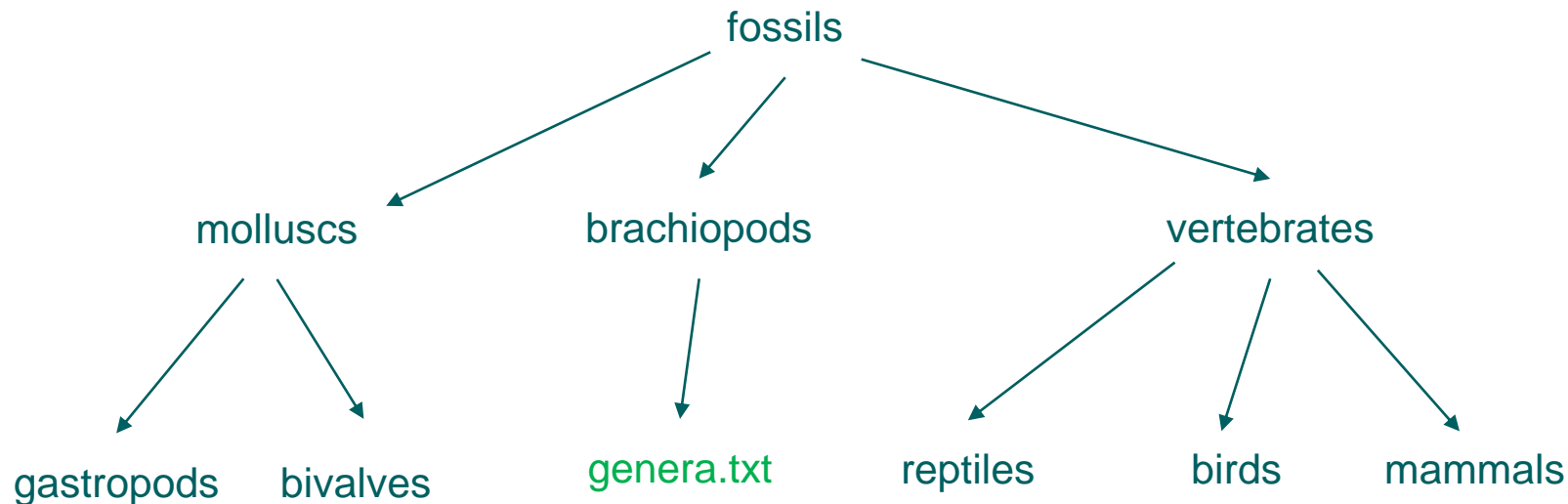
`\` : double quote escape

`'` : single quote escape



# Exercise!

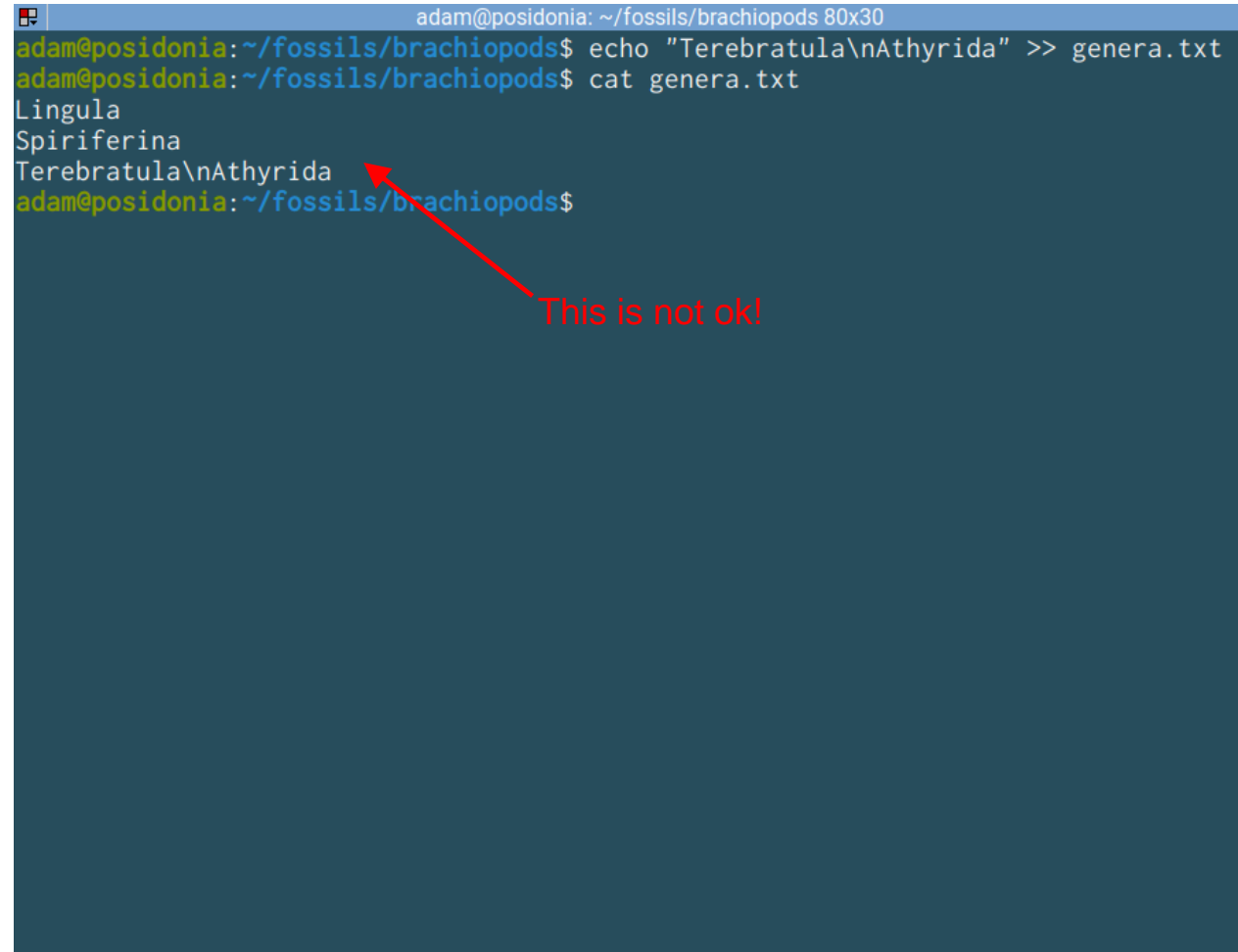
1. Use an `echo` statement to write the genus name “*Terebratula*” and “*Athyris*” into `fossils/brachiopods/genera.txt`, **use a newline escape between them!**
2. Then change directory to `brachiopods`.



```
echo_"Terebratula\nAthyrida">>genera.txt
```

## Appending to files

- It doesn't seem to work!
- Echo needs to know to replace the combination `\n` with the newline character!



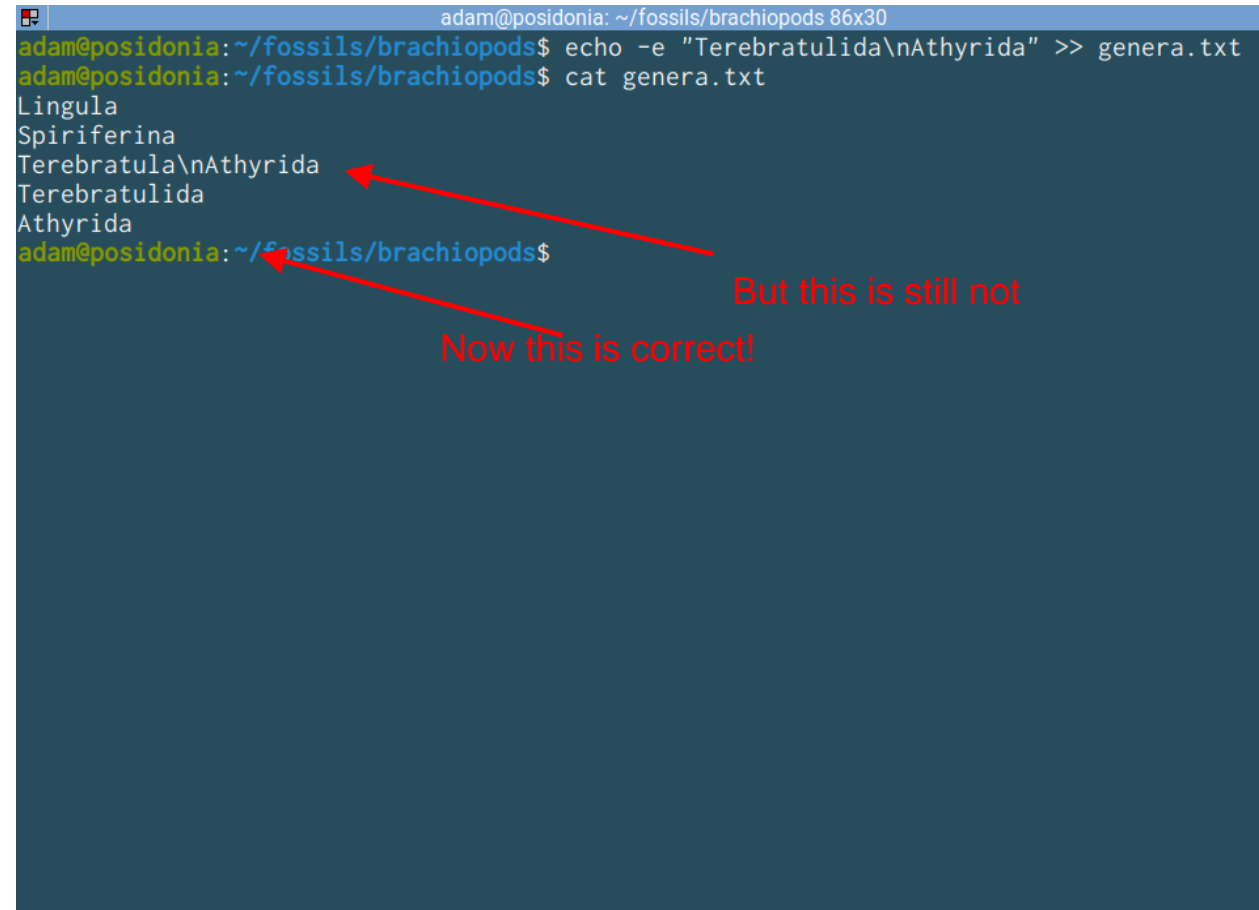
```
adam@posidonia: ~/fossils/brachiopods 80x30
adam@posidonia:~/fossils/brachiopods$ echo "Terebratula\nAthyrida" >> genera.txt
adam@posidonia:~/fossils/brachiopods$ cat genera.txt
Lingula
Spiriferina
Terebratula\nAthyrida
adam@posidonia:~/fossils/brachiopods$
```

This is not ok!

```
echo -e "Terebratula\nAthyrida" >> genera.txt
```

## Appending to files

- Use the `-e` option!
- Our file is messed up. Options:
  - 1. Redo our file
  - 2. Use an editor to correct
- Delete the bad line!
- Better, next time: go back in time.



The screenshot shows a terminal window with the following content:

```
adam@posidonia: ~/fossils/brachiopods 86x30
adam@posidonia:~/fossils/brachiopods$ echo -e "Terebratulida\nAthyrida" >> genera.txt
adam@posidonia:~/fossils/brachiopods$ cat genera.txt
Lingula
Spiriferina
Terebratula\nAthyrida
Terebratulida
Athyrida
adam@posidonia:~/fossils/brachiopods$
```

Two red arrows point from text annotations to the output. The first arrow points to the line `Terebratula\nAthyrida` and is accompanied by the text "But this is still not". The second arrow points to the line `Athyrida` and is accompanied by the text "Now this is correct!".

# Basic version control with Git

## and GitHub



# Why version control?

**Projects evolve in a non-linear way, especially programming projects.**

- Multiple people work on them, sometimes at the same time
- Recording the history of project development
- Working with many files
- Sharing code is necessary, we also need to know who changes what



# Difference between Git and GitHub?

## git

- Locally running application
- Operates with files in a local directory (repository)
- Works without a remotes!



## GitHub and GitLab

- Remote servers with copies of the repository

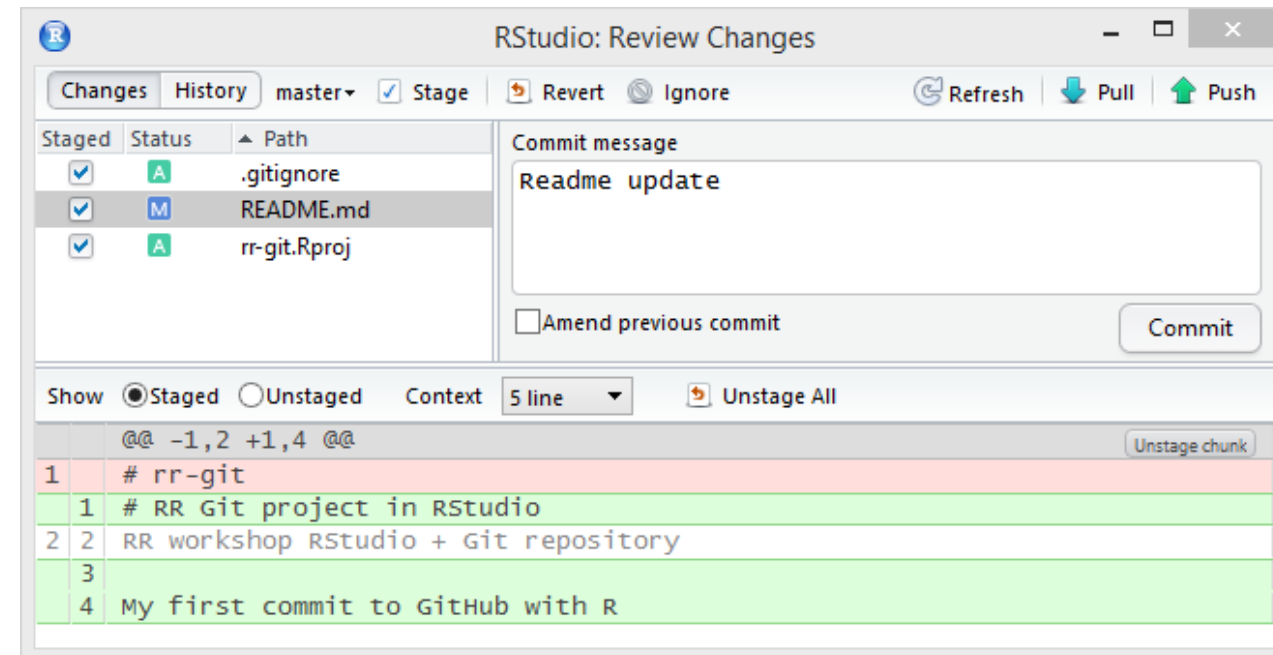


# Interfaces to git

**Git is a command line  
(console application)**

- The complete features are only available via the command line!
- Simplified graphical interfaces written for novices, embedded in IDEs
- These actually just translate the actions to the command line application -> Experiment!

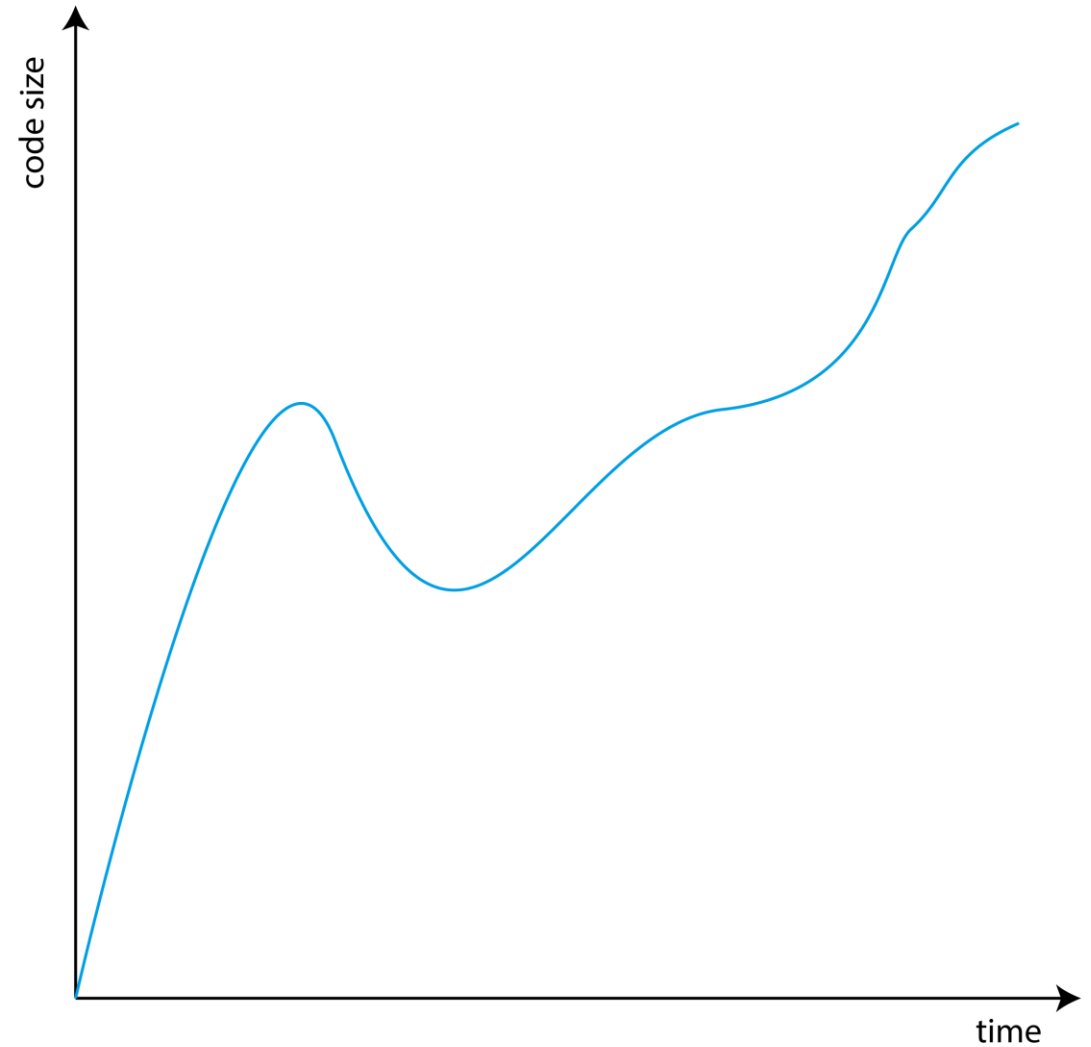
```
adam@posidonia: ~ 86x30
adam@posidonia:~$ git --version
git version 2.25.1
adam@posidonia:~$
```



# The basic use of git

**Record snapshots of how a project develops.**

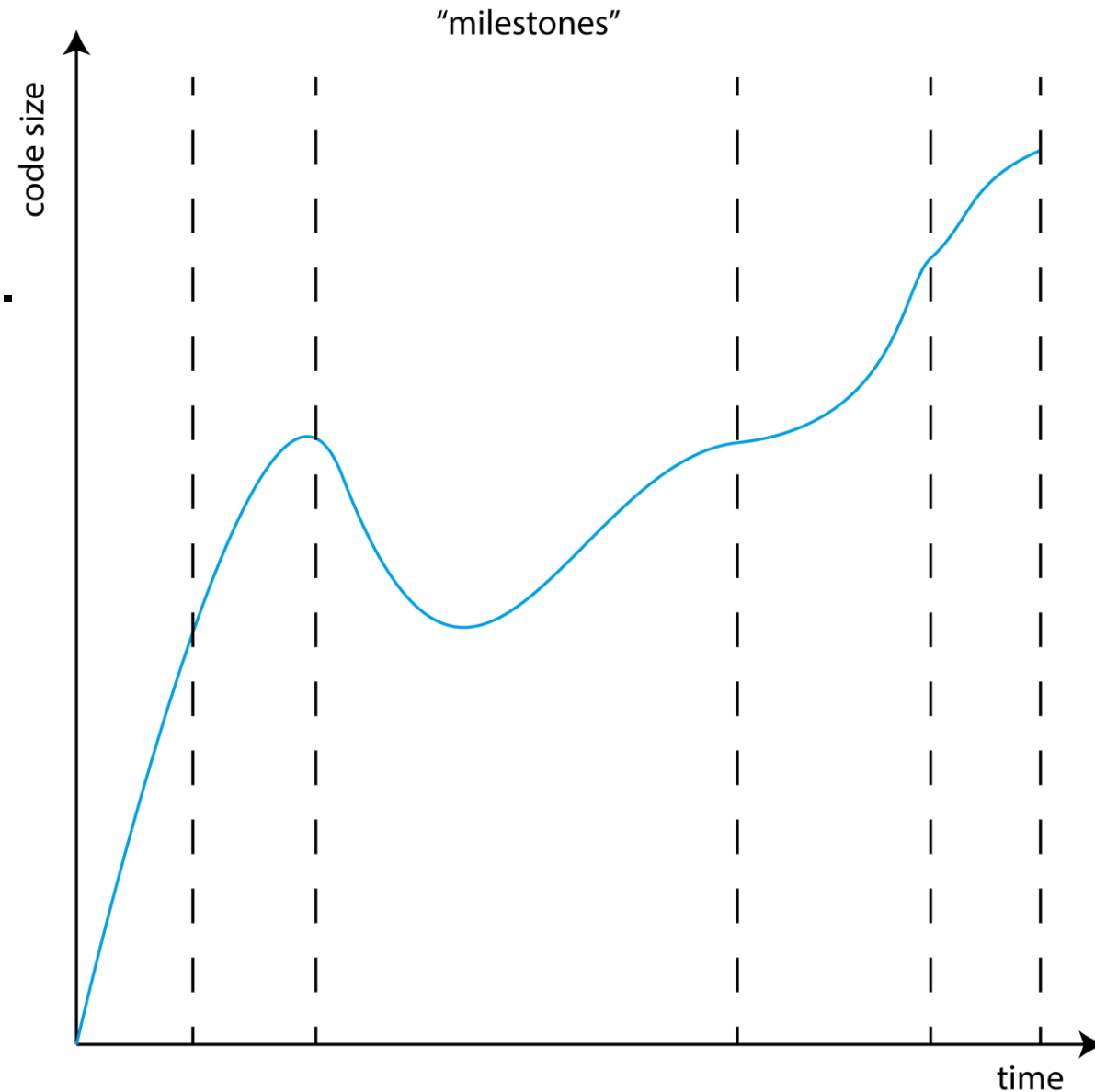
- Code develops in a non-linear, but continuous way, with lots of small changes:
  - Contents of files change
  - New files are added to the repository
  - Old files are delete from the respository



# The basic use of git

**Record snapshots of how a project develops.**

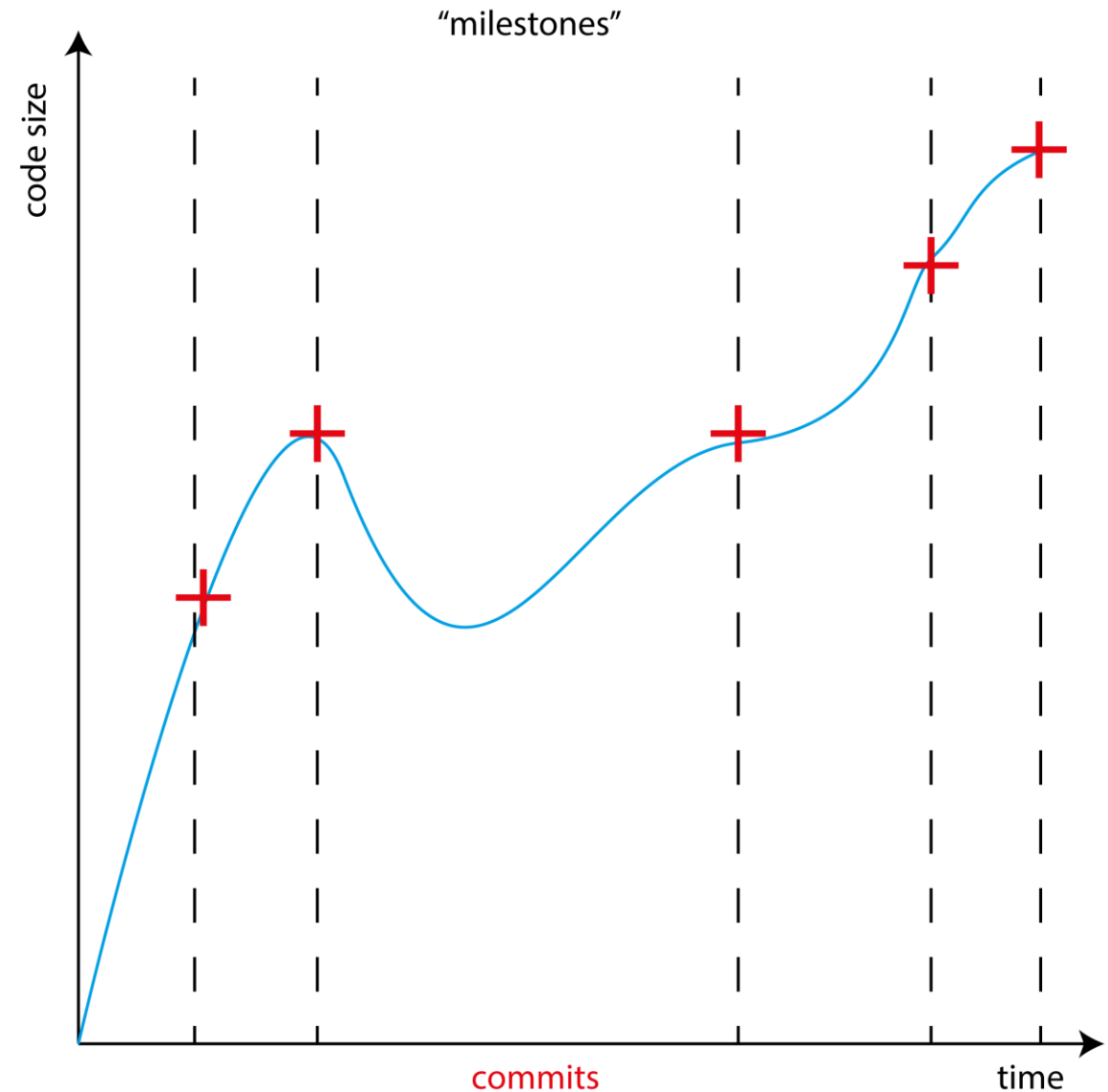
- Specific states of the code represent milestones:
  - Something works completely
  - Everything is cleaned up
  - Ready for further development
- In between these are transient states, when you are working on something but that is not yet done.



# The basic use of git

**Record snapshots of how a project develops.**

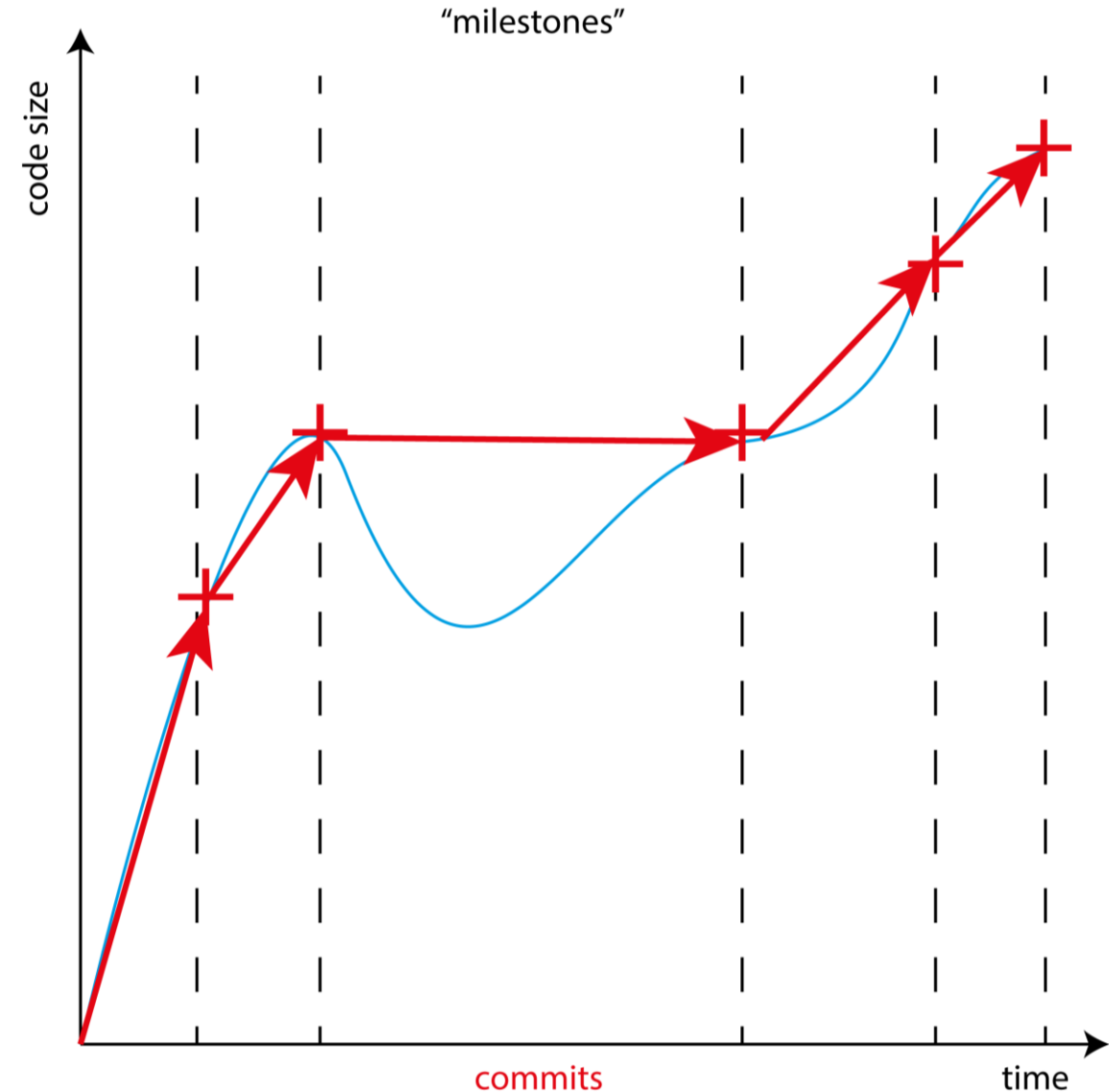
- These milestones can be saved and accessed at any time.
- These states are called as 'commits' in git's terminology



# The basic use of git

## Record snapshots of how a project develop

- Only the committed stages are recorded, the rest of the history is discarded
- The git repository is recorded as changes from one commit to the next



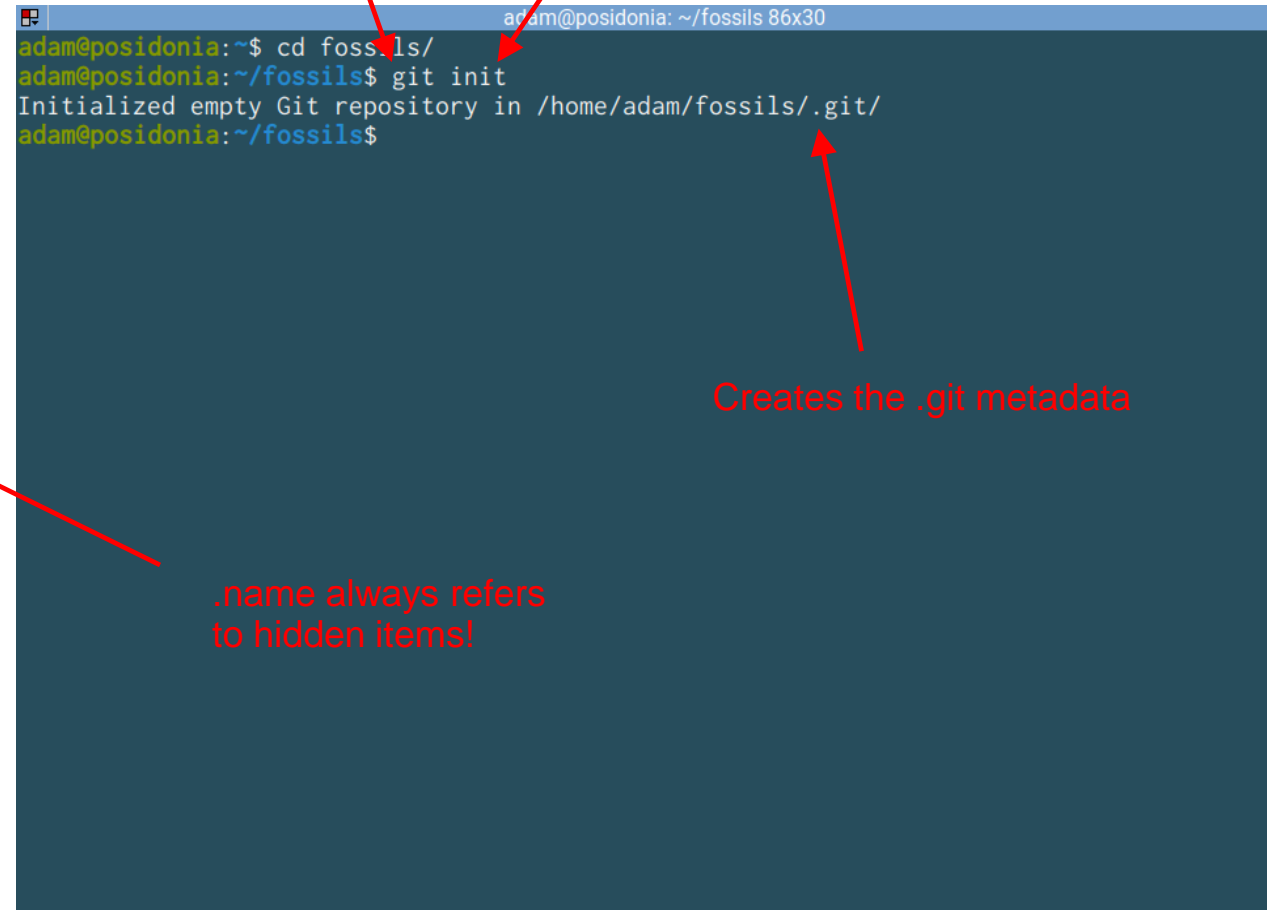
# git\_init

**Create a new git repository in current directory.**

- A git repository is a directory with git metadata in it.
- The git metadata are in the `.git` directory

Name of application

Command for the application



```
adam@posidonia: ~$ cd fossils/
adam@posidonia:~/fossils$ git init
Initialized empty Git repository in /home/adam/fossils/.git/
adam@posidonia:~/fossils$
```

Creates the .git metadata

.name always refers to hidden items!

The terminal screenshot shows a user named adam at a machine named posidonia. The user navigates to the directory ~/fossils and runs the command git init. The output shows that an empty Git repository has been initialized in the directory /home/adam/fossils/.git/. Red arrows point from the text annotations to the corresponding parts of the terminal output: one points to 'fossils/' in the prompt, another to 'git init', a third to the path '/home/adam/fossils/.git/' in the output, and a fourth to the '.git' part of the path in the output.



# ls -a

**List all files and directories in directory, including hidden items!**

- The double dot (..) represents a way to refer to the previous directory, as we have seen earlier
- The single dot (.) represents a way to refer to the current directory.
- **Note:** `cd brachiopods` and `cd ./brachiopods` are the same!

```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~$ cd fossils/
adam@posidonia:~/fossils$ git init
Initialized empty Git repository in /home/adam/fossils/.git/
adam@posidonia:~/fossils$ ls
brachiopods  molluscs  vertebrates
adam@posidonia:~/fossils$ ls -a
.  ..  brachiopods  .git  molluscs  vertebrates
adam@posidonia:~/fossils$
```

the .git metadata directory

Virtual directories . (dot) and .. (dot dot)

# git\_status

Show the status of the current repository

- A series of commits is called a 'branch'. Simple repos use only one. There is always a current one
- Git has detected that there are things in the repo that are not registered.
- **Git can only detect files. Empty directories are not recorded!**

The image shows a terminal window with the command `git status` executed. Red arrows point from text labels to specific parts of the terminal output:

- Name of application** points to `git` in the command.
- Command for the application** points to `status` in the command.
- Name of the current "branch"** points to `master` in the output "On branch master".
- We have no commits yet** points to the line "No commits yet".
- brachiopods/** points to the untracked file path.
- NOTE: In many cases, git literally tells you what to do.** points to the instruction "(use 'git add' to track)".

```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  brachiopods/

nothing added to commit but untracked files present (use "git add" to track)
adam@posidonia:~/fossils$
```

# Staging

## The preparation of a commit

- Commits are permanent, or are difficult to remove once done, so we have tools to make sure that they are ok
- Changes first have to be staged, before committing. This allows us to include only specific changes in the commit, and to make sure that we are doing things ok.

# Staging and commit (Airport)

## Initial boarding pass control vs. boarding

- If you go through security you are staged to fly. You are expected to be on the plane, but you can still leave.
- If you board the plane and the cabin doors are closed, you are committed to a flight.

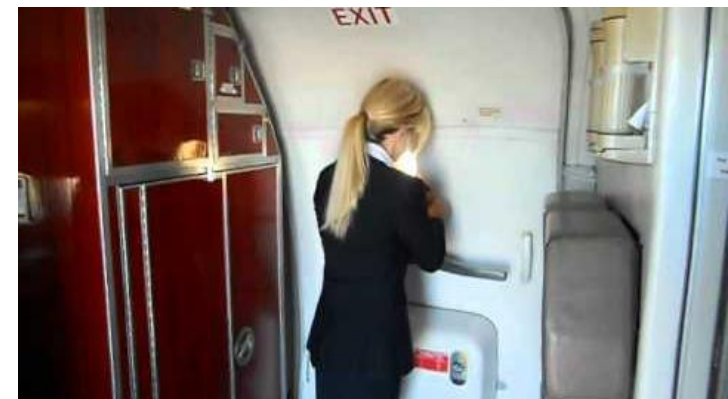
Getting staged



In the staged area, waiting to be committed



The commit

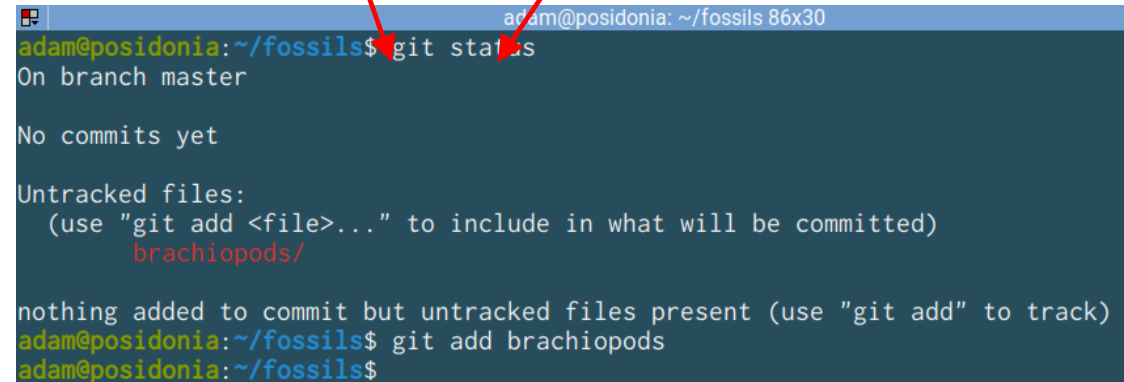


`git_add_<path>`

**Stage the target file or directory.**

- Frequently this is an entire directory, including . (dot)
- If successful does not return anything, has to be checked with git status

Name of application      Command for the application



```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    brachiopods/

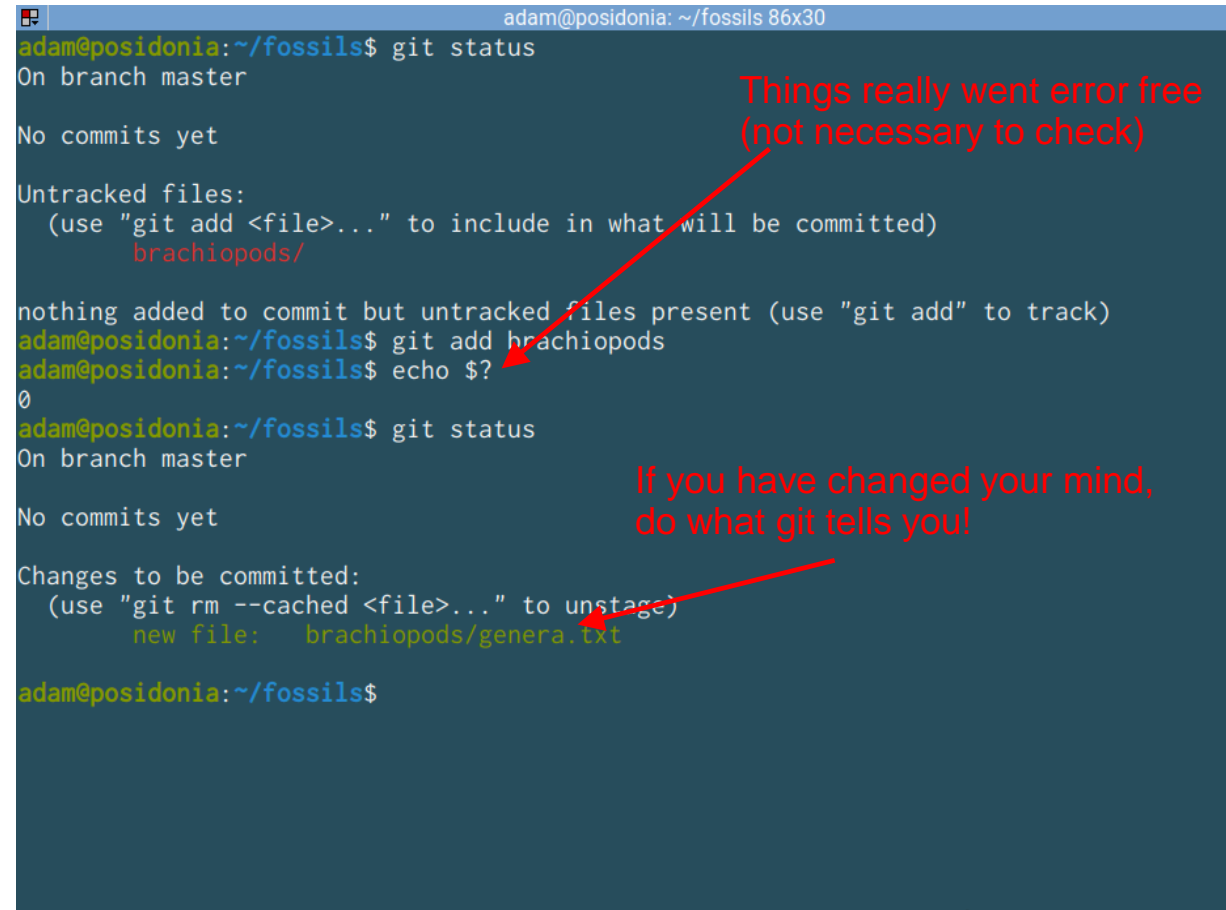
nothing added to commit but untracked files present (use "git add" to track)
adam@posidonia:~/fossils$ git add brachiopods
adam@posidonia:~/fossils$
```

The terminal screenshot shows the execution of `git status` and `git add brachiopods`. A red arrow points from the text 'Name of application' to the word 'fossils' in the terminal's title bar. Another red arrow points from the text 'Command for the application' to the word 'git' in the `git status` command.

# git\_status (again)

## Show status of repo

- There is just one file here which git finds.
- The file is now stages to be committed.



```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    brachiopods/

nothing added to commit but untracked files present (use "git add" to track)
adam@posidonia:~/fossils$ git add brachiopods
adam@posidonia:~/fossils$ echo $?
0
adam@posidonia:~/fossils$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   brachiopods/genera.txt

adam@posidonia:~/fossils$
```

Things really went error free  
(not necessary to check)

If you have changed your mind,  
do what git tells you!

# git\_commit\_-m\_<message>

## First use not permitted without credentials!

- You need to provide a user name and an email address with the git config command

```
File Machine View Input Devices Help
MINGW64/c/Users/Adam/fossils

Adam@Teaching MINGW64 ~/fossils (master)
$ git commit -m "First file added"
Author identity unknown

*** please tell me who you are.

Run

    git config --global user.email "you@example.com"
    git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'Adam@Teaching.(none)')

Adam@Teaching MINGW64 ~/fossils (master)
$ |
```

# git\_config\_--global\_<what>\_<value>

## Configuring git

- user.name and user.email
- --global sets this for all your local git repositories
- Now you are ready to commit

```
File Machine View Input Devices Help
MINGW64/c/Users/Adam/fossils

Adam@Teaching MINGW64 ~/fossils (master)
$ git commit -m "First file added"
Author identity unknown

*** Please tell me who you are.

Run

    git config --global user.email "you@example.com"
    git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'Adam@Teaching.(none)')

Adam@Teaching MINGW64 ~/fossils (master)
$ git config --global user.email "adam.kocsis@outlook.com"

Adam@Teaching MINGW64 ~/fossils (master)
$ git config --global user.name "adamkocsis"

Adam@Teaching MINGW64 ~/fossils (master)
$
```



# git\_commit\_-m\_<message>

## Now create a new commit

- Provide a message in **quotes!**  
This is the human readable description of what changed.
- Every commit gets a unique 'hash', a random set of characters that are used to identify unambiguously identify the commit

The screenshot shows a terminal window with the following text and annotations:

```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ git commit -m "First file added."
[master (root-commit) 6c61587] First file added.
1 file changed, 4 insertions(+)
create mode 100644 brachiopods/genera.txt
adam@posidonia:~/fossils$
```

Annotations with red arrows:

- The message you provided**: Points to the message in quotes: "First file added."
- The beginning of the hash of the commit.**: Points to the first few characters of the commit hash: 6c61587.
- Four new lines are added**: Points to the line "1 file changed, 4 insertions(+)" in the output.
- The affected files**: Points to the file path "brachiopods/genera.txt" in the output.
- branch**: Points to the word "master" in the output.

# git\_status (yet again)

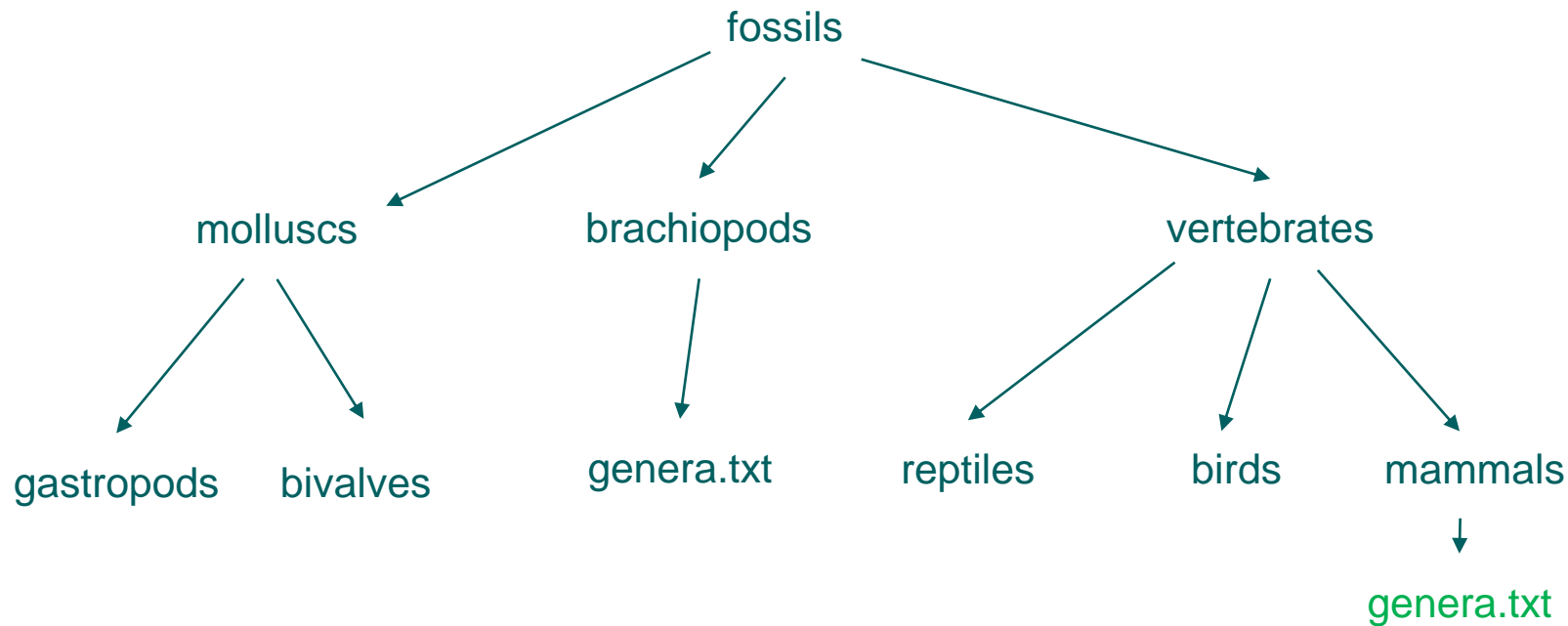
**Nothing to be done.**

- Create two new files

```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ git commit -m "First file added."
[master (root-commit) 6c6158e] First file added.
 1 file changed, 4 insertions(+)
 create mode 100644 brachiopods/genera.txt
adam@posidonia:~/fossils$ git status
On branch master
nothing to commit, working tree clean
adam@posidonia:~/fossils$
```

# Exercise!

1. Create a new file `genera.txt` in the `mammals` directory, and put the names of 3 mammalian genera in it!
2. Stage and commit the changes!



# My solution

Nothing to be done.

```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ echo -e "Mustela\nHomo\nPanthera" > ./vertebrates/mammals/genera.txt
adam@posidonia:~/fossils$ cat vertebrates/mammals/genera.txt
Mustela
Homo
Panthera
adam@posidonia:~/fossils$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    vertebrates/

nothing added to commit but untracked files present (use "git add" to track)
adam@posidonia:~/fossils$ git add .
adam@posidonia:~/fossils$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   vertebrates/mammals/genera.txt

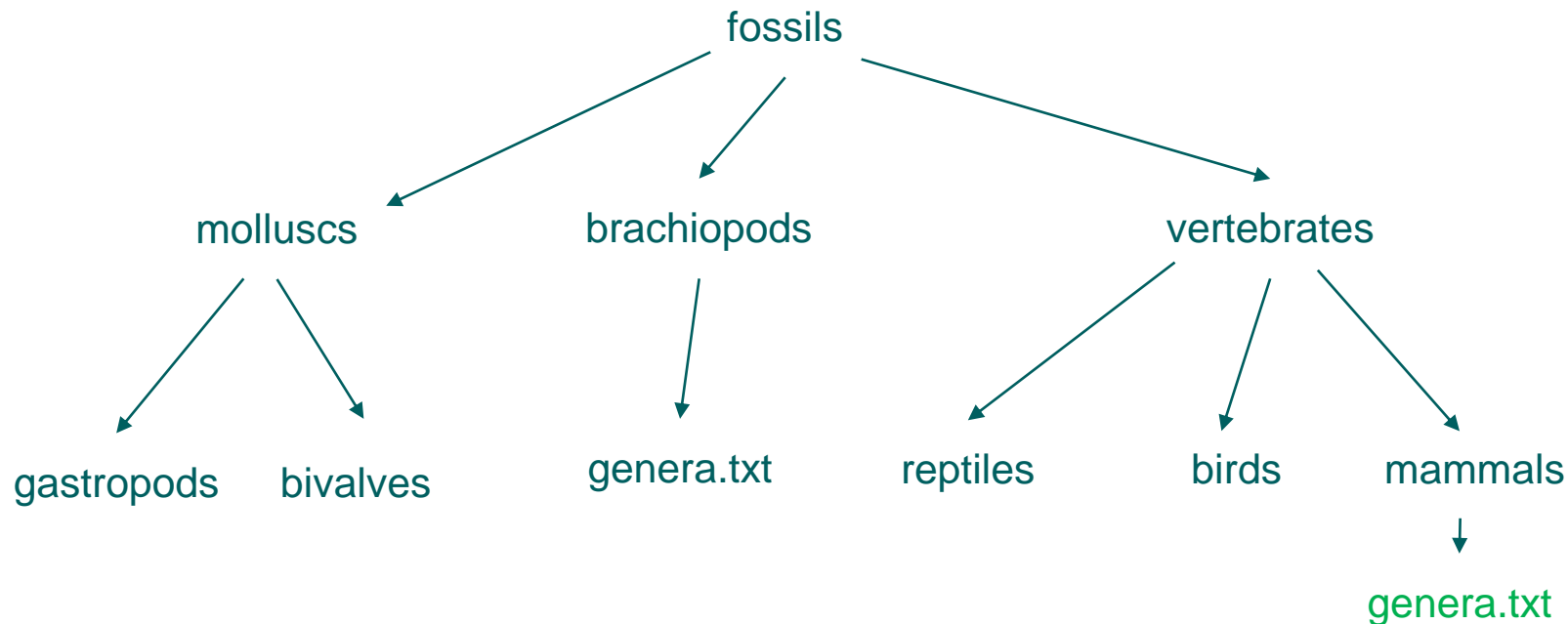
adam@posidonia:~/fossils$ git commit -m "added vertebrate genera"
[master 510177f] added vertebrate genera
1 file changed, 3 insertions(+)
create mode 100644 vertebrates/mammals/genera.txt
adam@posidonia:~/fossils$
```

Contents of the new file

Add everything you find in current directory.

# Exercise!

1. Create a new file `genera.txt` in the `birds` directory, and put the names of 2 bird genera in it!
2. Add another genus to the mammals.
3. Try to commit only the birds!



# My solution

## 1. Make the changes.

```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ echo -e "Pica\nTurdus" > "vertebrates/birds/genera.txt"
adam@posidonia:~/fossils$ cat vertebrates/birds/genera.txt
Pica
Turdus
adam@posidonia:~/fossils$ echo "Talpa" >> vertebrates/mammals/genera.txt
adam@posidonia:~/fossils$ cat vertebrates/mammals/genera.txt
Mustela
Homo
Panthera
Talpa
adam@posidonia:~/fossils$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   vertebrates/mammals/genera.txt
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        vertebrates/birds/
no changes added to commit (use "git add" and/or "git commit -a")
adam@posidonia:~/fossils$
```

← Add birds

← Added another mammal

← Change in already committed file

← New entries to be added

# My solution

## 2. Stage only the birds.

```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ cat vertebrates/mammals/genera.txt
Mustela
Homo
Panthera
Talpa
adam@posidonia:~/fossils$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   vertebrates/mammals/genera.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        vertebrates/birds/

no changes added to commit (use "git add" and/or "git commit -a")
adam@posidonia:~/fossils$ git add vertebrates/birds
adam@posidonia:~/fossils$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   vertebrates/birds/genera.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   vertebrates/mammals/genera.txt

adam@posidonia:~/fossils$
```

**Staged.** (arrow pointing to new file: vertebrates/birds/genera.txt)

**Not staged!** (arrow pointing to modified: vertebrates/mammals/genera.txt)

# My solution

## 3. Stage only the birds.

```
adam@posidonia: ~/fossils 86x30
Talpa
adam@posidonia:~/fossils$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   vertebrates/mammals/genera.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        vertebrates/birds/

no changes added to commit (use "git add" and/or "git commit -a")
adam@posidonia:~/fossils$ git add vertebrates/birds
adam@posidonia:~/fossils$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   vertebrates/birds/genera.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   vertebrates/mammals/genera.txt

adam@posidonia:~/fossils$ git commit -m "added bird genera"
[master b53f2f9] added bird genera
1 file changed, 2 insertions(+)
create mode 100644 vertebrates/birds/genera.txt
adam@posidonia:~/fossils$
```

Nothing happened to mammals!



# git\_restore\_<path>

## Discarding changes from previous commit

- We can commit the new mammal or discard it.
- You can correct unintended changes with this.
- What about even older changes?

Again, git literally tells you your options

```
adam@posidonia: ~/fossils 86x30
adam@posidonia:~/fossils$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   vertebrates/mammals/genera.txt

no changes added to commit (use "git add" and/or "git commit -a")
adam@posidonia:~/fossils$ git restore vertebrates/mammals/genera.txt
adam@posidonia:~/fossils$ cat vertebrates/mammals/genera.txt
Mustela
Homo
Panthera
adam@posidonia:~/fossils$
```

The file is restored to the state before the changes, what is in the commit.

# GitHub

and GitHub

# GitHub

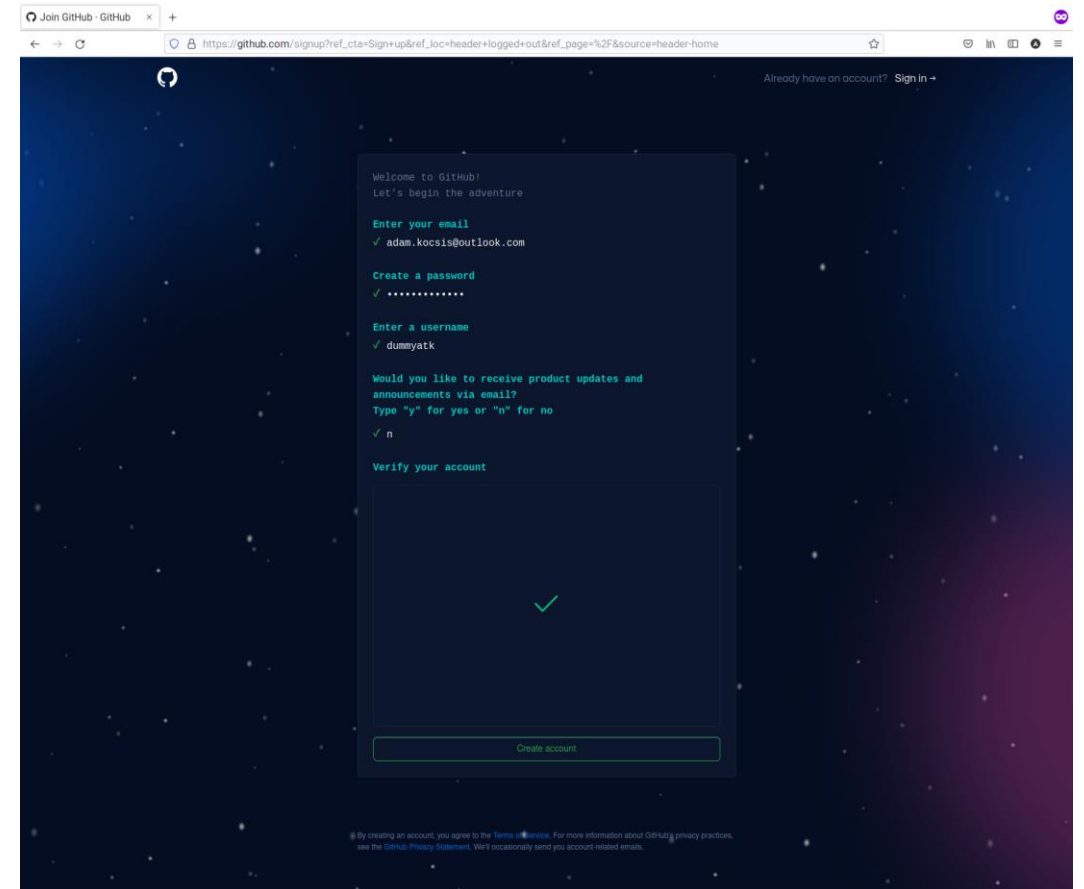
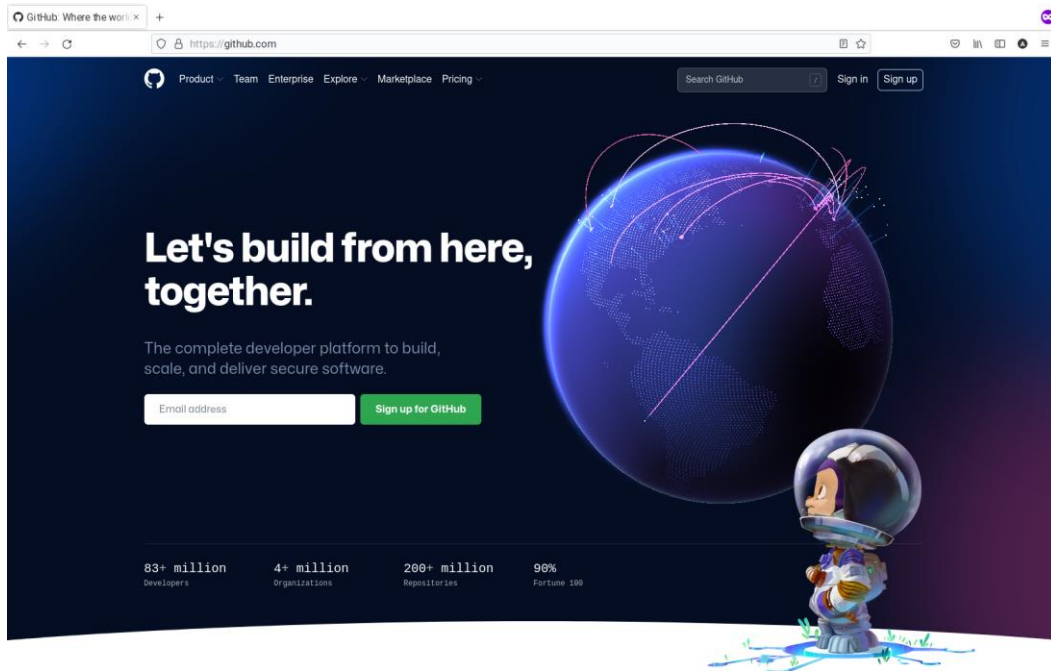
## Where the world builds software (2008-)

- Open source software development platform, places to store and share git repositories
- Currently owned by Microsoft
- Applications, packages, plugins, webpages and many more!
- Free and private repositories.



# GitHub

Sign up if you haven't yet!



# GitHub - Dashboard

The screenshot shows the GitHub dashboard interface. At the top, there's a navigation bar with a search bar, links for Pull requests, Issues, Marketplace, and Explore. On the right side of the navigation bar, there are icons for Notifications (a bell) and Settings (a gear). Two red arrows point from the text labels 'Notifications' and 'Settings' to these respective icons. The main content area is divided into several sections: 'Create your first project' with buttons for 'Create repository' and 'Import repository'; 'Recent activity'; 'The home for all developers — including you.' which includes 'Start writing code' and 'Use tools of the trade'; and a grid of cards for 'Start a new repository', 'Create your profile README', 'Contribute to an existing repository', 'Write code in your web browser', 'Install a powerful code editor', and 'Set up your local dev environment'. On the right side, there are two floating cards: 'GitHub Copilot' and 'PRIVACY STATEMENT UPDATES Adding web cookies for enterprise users'. The bottom right corner shows 'Latest changes'.

# GitHub - Dashboard

Access your repos

Settings

The screenshot shows the GitHub dashboard for a user named 'dummyatk'. The user menu is open, showing options like 'Your repositories', 'Your codespaces', 'Your projects', 'Your stars', 'Your gists', 'Upgrade', 'Feature preview', 'Help', 'Settings', and 'Sign out'. A red arrow points from the text 'Access your repos' to the 'Your repositories' option in the menu. Another red arrow points from the text 'Settings' to the 'Settings' option in the menu. The dashboard itself features a 'Create your first project' section, a 'Recent activity' section, and a 'The home for all developers' section with various tools and guides. The 'Your repositories' section is highlighted in blue.

GitHub

Search or jump to...

Pull requests Issues Marketplace Explore

Create your first project

Ready to start building? Create a repository for a new idea or bring over an existing repository to keep contributing to it.

Create repository Import repository

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

The home for all developers — including you.

Welcome to your personal dashboard, where you can find an introduction to how GitHub works, tools to help you build software, and help merging your first lines of code.

Start writing code

Start a new repository

Collaborate on code with others and track your work in a repository.

Create a new repository

Create your profile README

Create a file in a repository that tells the GitHub community who you are.

Create a README

Contribute to an existing repository

Find repos that need your help

GitHub Copilot

Get suggestions for lines of code and entire functions in real-time

Learn more about Copilot

PRIVACY STATEMENT UPDATES

Adding web cookies to enterprise users

In order to better reach and improve the experience for enterprise users, we are adding non-essential web cookies to certain subdomains that specifically market our products to businesses. This change is only on subdomains that reach enterprise customers, and all other GitHub subdomains will continue to operate as-is.

Learn more

Latest changes

# GitHub – Creating a new repo

Make a new repo

The screenshot shows the GitHub interface for a user named 'dummyatk'. At the top, there's a navigation bar with 'Your Repositories' and a search bar. Below this, the user's profile is displayed, including a circular profile picture of a man and the username 'dummyatk'. To the right of the profile, there are tabs for 'Overview', 'Repositories', 'Projects', 'Packages', and 'Stars'. Below the tabs is a search bar labeled 'Find a repository...' and filters for 'Type', 'Language', and 'Sort'. A green 'New' button with a repository icon is located to the right of these filters. A red arrow points from the text 'Make a new repo' to this 'New' button. Below the filters, a message states 'dummyatk doesn't have any public repositories yet.' At the bottom of the profile section, there is an 'Edit profile' button and a note 'Joined 7 minutes ago'. The footer of the page contains the GitHub logo, copyright information, and various links like 'Terms', 'Privacy', 'Security', etc.




# GitHub – Creating a new repo



## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \*

 dummyatk

Repository name \*

fossils

Great repository names are short and memorable. Need inspiration? How about [shiny-meme?](#)

Description (optional)

Just an exercise.

☒ Public

Anyone on the internet can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: None

 You are creating a public repository in your personal account.

Create repository

Usually the same as the local directory.

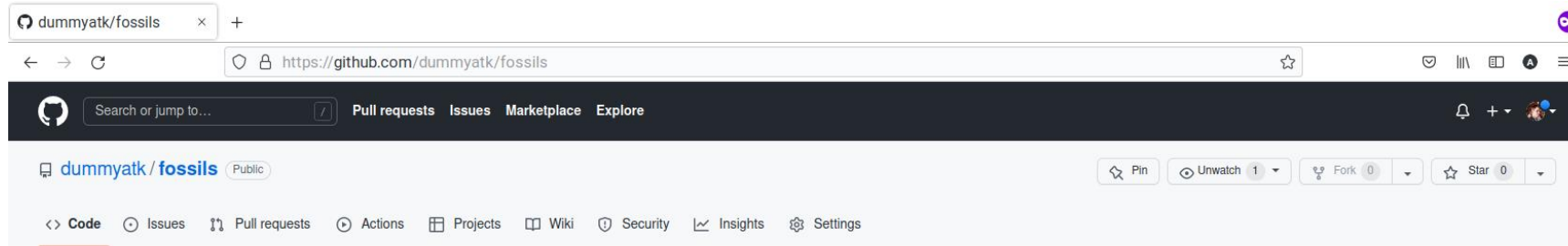
Do not change these. You will copy files over from your local repo.

Things for others!

Create!



# GitHub – The fresh empty repo



This is what you want

**Quick setup — if you've done this kind of thing before**

or ☐ HTTPS ☐ SSH

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

---

**...or create a new repository on the command line**

```
echo "# fossils" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/dummyatk/fossils.git
git push -u origin main
```

---

**...or push an existing repository from the command line**

```
git remote add origin https://github.com/dummyatk/fossils.git
git branch -M main
git push -u origin main
```

---

**...or import code from another repository**

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

We have already done this mostly

# GitHub – Add new remote

The name of the new remote. You can refer to it from now on using this name!

The URL of the remote. This is used to identify the remote on the web.

You are registering a new remote

Command: you want to make changes of how your local repository is connected to remotes

Application name

**...or push an existing repository from the command line**

```
git remote add origin https://github.com/dummyatk/fossils.git
git branch -M main
git push -u origin main
```

# GitHub – Rename current branch to main

Application name

Command: you want to do  
things with branches

Move all contents of current branch  
to

The name of the branch (new)

**...or push an existing repository from the command line**

```
git remote add origin https://github.com/dummyatk/fossils.git
git branch -M main
git push -u origin main
```

**For political reasons, GitHub does not allow the use of the name master, hence this extra step.**

# GitHub – Pushing contents of branch to remote

Application name

Command: you want copy  
contents from local to  
remote

Set the default remote and branch

Remote to copy material to

Which branch to push?

**...or push an existing repository from the command line**

```
git remote add origin https://github.com/dummyatk/fossils.git  
git branch -M main  
git push -u origin main
```

**GitHub will ask for your credentials**

# GitHub – Executing this and signing in on windows

Note branch name change

Most interactive sign  
in option available on  
Windows

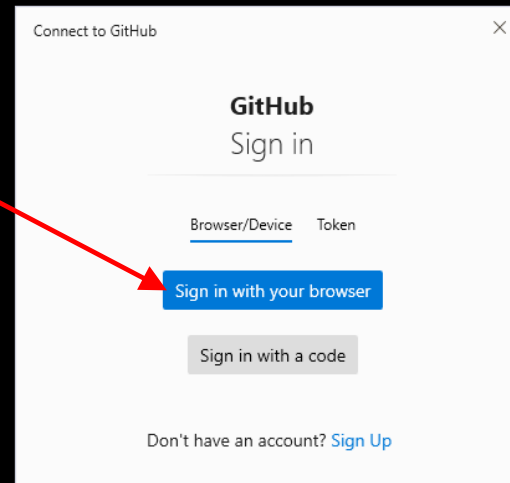
File Machine View Input Devices Help

MINGW64: c:/Users/Adam/fossils

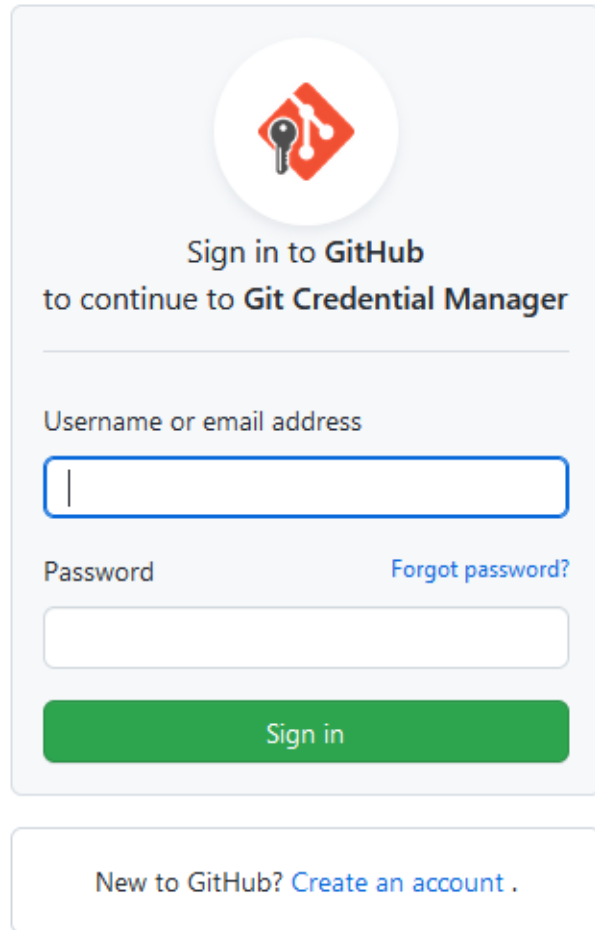
```
Adam@Teaching MINGW64 ~/fossils (master)
$ git remote add origin https://github.com/dummyatk/fossils.git

Adam@Teaching MINGW64 ~/fossils (master)
$ git branch -M main

Adam@Teaching MINGW64 ~/fossils (main)
$ git push -u origin main
```



# GitHub – Executing this and signing in on windows



Sign in to GitHub  
to continue to Git Credential Manager

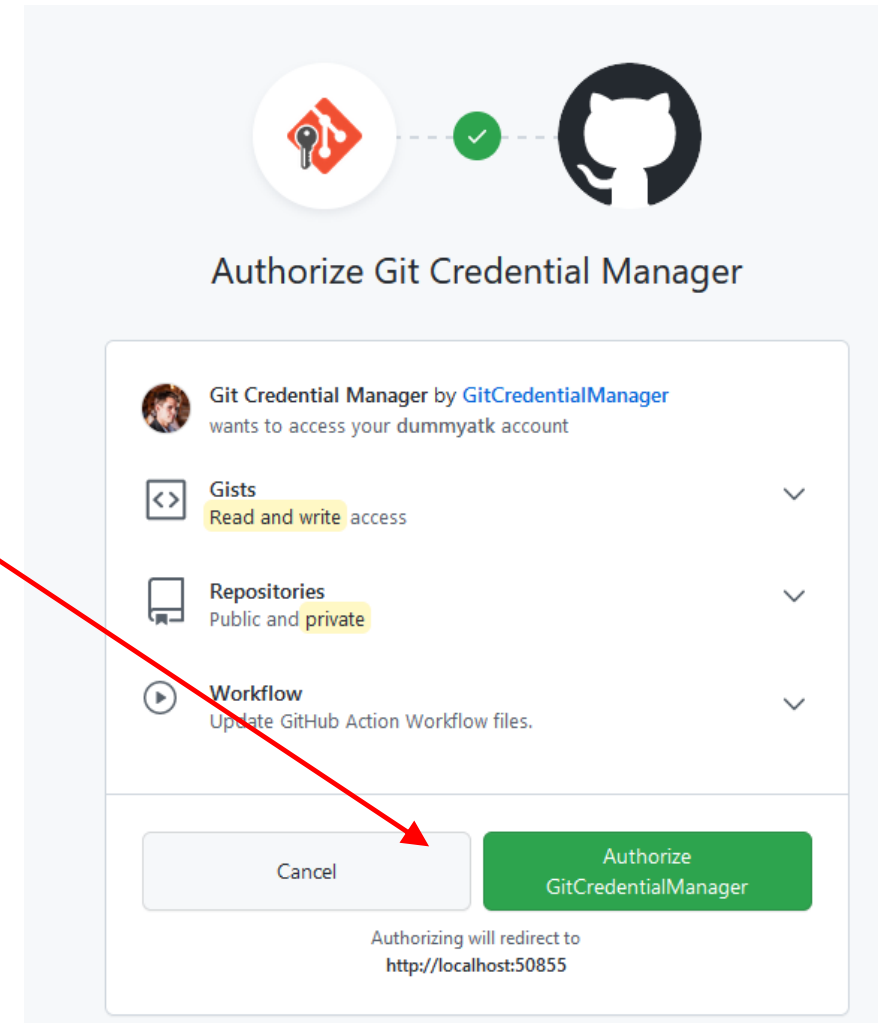
Username or email address

Password [Forgot password?](#)




[Sign in](#)


New to GitHub? [Create an account](#).




This is what you want



Authorize Git Credential Manager

 Git Credential Manager by [GitCredentialManager](#)  
wants to access your dummyatk account

-  **Gists**  
Read and write access
-  **Repositories**  
Public and private
-  **Workflow**  
Update GitHub Action Workflow files.

[Cancel](#) [Authorize GitCredentialManager](#)

Authorizing will redirect to  
<http://localhost:50855>

# GitHub – Successful push

Transfer stats

New branch main is  
created on remote

And is now in sync with  
local

```
File  Machine  View  Input  Devices  Help
MINGW64:/c/Users/Adam/fossils

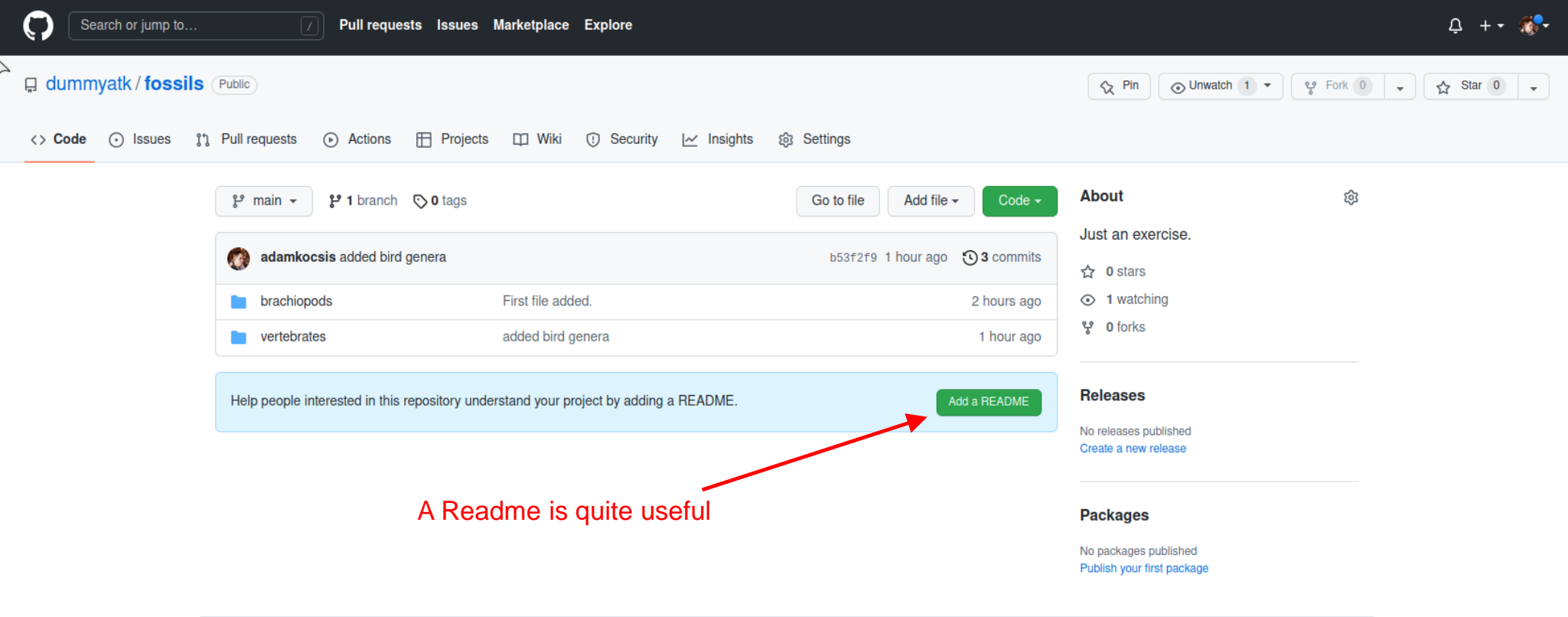
Adam@Teaching MINGW64 ~/fossils (master)
$ git remote add origin https://github.com/dummyatk/fossils.git

Adam@Teaching MINGW64 ~/fossils (master)
$ git branch -M main

Adam@Teaching MINGW64 ~/fossils (main)
$ git push -u origin main
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (14/14), 1.03 KiB | 1.03 MiB/s, done.
Total 14 (delta 0), reused 14 (delta 0), pack-reused 0
To https://github.com/dummyatk/fossils.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

Adam@Teaching MINGW64 ~/fossils (main)
$ |
```

# GitHub – Successful push



The screenshot shows the GitHub interface for the repository 'dummyatk/fossils'. The repository is public and has 1 branch (main) and 0 tags. A recent commit by 'adamkocsis' is shown, titled 'added bird genera', with a commit hash of 'b53f2f9' and 3 commits. The commit message is 'added bird genera'. Below the commit, there are two files listed: 'brachiopods' (First file added, 2 hours ago) and 'vertebrates' (added bird genera, 1 hour ago). A blue banner at the bottom of the commit section prompts the user to 'Help people interested in this repository understand your project by adding a README.' with a green 'Add a README' button. A red arrow points from the text 'A Readme is quite useful' to the 'Add a README' button. The right sidebar shows the repository's statistics: 0 stars, 1 watching, and 0 forks. The 'Releases' section shows 'No releases published' with a link to 'Create a new release'. The 'Packages' section shows 'No packages published' with a link to 'Publish your first package'.

Search or jump to... Pull requests Issues Marketplace Explore

dummyatk / fossils Public

Pin Unwatch 1 Fork 0 Star 0

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags

Go to file Add file Code

adamkocsis added bird genera b53f2f9 1 hour ago 3 commits

|             |                   |             |
|-------------|-------------------|-------------|
| brachiopods | First file added. | 2 hours ago |
| vertebrates | added bird genera | 1 hour ago  |

Help people interested in this repository understand your project by adding a README. Add a README

About Just an exercise. 0 stars 1 watching 0 forks

Releases No releases published Create a new release

Packages No packages published Publish your first package

© 2022 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

A Readme is quite useful



# Writing a Readme

- Default format is **markdown (later)**
- You can work on files using GitHub's interface
- Save the defaults, by clicking on the green button
- Note that you are technically creating a new commit!

The screenshot shows the GitHub web interface for creating a new file. The repository is 'dummyatk / fossils'. The file being created is 'README.md' in the 'main' branch. The editor shows the following content:

```
1 # fossils
2 Just an exercise.
3
```

Red arrows and text annotations highlight key features:

- File name: .md is for markdown.** Points to the 'README.md' filename in the editor header.
- Some default content, we will visit this again.** Points to the default markdown content in the editor.
- Default commit message.** Points to the 'Commit new file' button in the commit dialog.

The commit dialog shows the following options:

- Commit directly to the `main` branch. (Selected)
- Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Buttons: **Commit new file** (green), **Cancel** (red).

# GitHub – Changing the remote

The screenshot shows the GitHub interface for the repository 'dummyatk/fossils'. The repository is public and has 1 branch (main) and 0 tags. The commit history shows a recent commit 'a4a30cd' by 'dummyatk' with the message 'Create README.md'. The file 'README.md' is listed in the commit details. The README content is displayed below, showing the title 'fossils' and the text 'Just an exercise.'.

Annotations:

- The very last commit's hash**: Points to the commit hash 'a4a30cd'.
- The message of the last commit that modified the file**: Points to the commit message 'Create README.md'.
- Readme file now added!**: Points to the 'README.md' file in the commit details.
- Beautifully rendered markdown document**: Points to the rendered README content.

# git\_pull\_<remote>\_<branch>

## Pull changes from remote

- Just because you changed something on the remote server does not make things magically appear locally
- You have to pull the contents of the remote to have the new file that you just created!

File Machine View Input Devices Help

MINGW64:/c/Users/Adam/fossils

```
Adam@Teaching MINGW64 ~/fossils (main)
$ git pull origin main
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 715 bytes | 55.00 KiB/s, done.
From https://github.com/dummyatk/fossils
 * branch                main          -> FETCH_HEAD
    b53f2f9..a4a30cd      main          -> origin/main
Updating b53f2f9..a4a30cd
Fast-forward
 README.md | 2 ++
 1 file changed, 2 insertions(+)
 create mode 100644 README.md

Adam@Teaching MINGW64 ~/fossils (main)
$ cat README.md
# fossils
Just an exercise.

Adam@Teaching MINGW64 ~/fossils (main)
$ |
```