Workflow NASC / ložiska

c3d mp2rage\_location\_r.nii.gz -replace 2 1 3 1 4 1 5 0 6 0 7 0 8 0 -o CCtoC3.nii.gz

c3d mp2rage\_fujimoto\_r\_lesion.nii.gz -replace 1 9 -o lesions.nii.gz

c3d CCtoC3.nii.gz lesions.nii.gz -add -o nasc\_lesions.nii.gz

c3d mp2rage\_T1Map\_r.nii.gz nasc\_lesions.nii.gz -lstat | sed 's/\./,/g' > segmentation\_results\_comma.csv

Opening txt file with VS code: code name\_of\_the\_file.txt

COMMAND LINE FOR BEGINNERS

Up- and down = previous commands

Ctrl + l = clears everything right up??

Ctrl + c = cancels command, goes on other line and not execute it

Ctrl + r = searches the command

Ctrl + d = closes the whole terminal

Commands:

--help

Whoami

**Commands for navigating – IMPORTANT!!!:**

pwd = prints the working direktory

ls = lists out the content of direktory or folder

ls -a (shows also hidden files)

ls -l (longlisting - shows extra informations – permissions, owners….)

cd = changing directories

* If I just type cd, it takes me into my home direktory
* cd Downloads
* cd .. = going back one level
* cd / going to the root
* cd – going to the last folder I was in (not the parent, but the last)

start = opening a file or a folder from the terminal in a GUI

**Commands for creating and modifying files and directories:**

mkdir =make directory

touch = make file (txt.) and possibly other files than txt

rm = deleting / removing a file

rm -r nebo rm -rf = deleting / removing a direktory **!!! be very aware what you are removing !!!**

cp = copying a file somewhere (cp file1.txt directory/file1.txt)

mv = moving a file somewhere (mv file1.txt directory/file1.txt) ; mv directory/file1.txt info.txt (přesune soubor jinam + přejmenuje ho)

open = open file v GUI ????? (nefunguje???)

nano = terminal-based text editor

--------------------------------------------------------------------------------------------------------------------------------------

**Showing content**

cat = displays

cat > writes into file

cat >>

cat -n = shows line number

less = shows content?

head = shows first ten lines

head -n = shows first n lines

tail a tail – n shows 10 / n last lines

echo = displaying text or content

**Search /find**

grep = searches for content, lines… (grep "line 22" file1.txt)

find = finds files in directories based on specified conditions (find . -name "file1.txt")

!!!!!!!! touch file-{001..100}.txt =creates 100 files from 001 to 100 in couple of miliseconds😉

(find . -name "file-\*")

(find . -name „file-\*.txt" -delete) = delets all (100) files again (CAVE!!!)

**Piping – redirectioning, transfering !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!**

Piping used > angle brackett

Example:

find . -name "file-\*" > output.txt (přesune všech 100 souborů do filu output)

cat output.txt (ukáže, co je v output)

**Symlink = special type of file that points to different file or directory (basically a shortcut)**

ln -s (name\_of\_directory) name\_of\_symlink OR mklink

?? it is not a copy, but a shortcut ??

**Tarballs = compressed files**

tar -cvzf source.tar.gz source = creates compressed file

tar -tzvf source.tar.gz = shows what is in compressed file

tar -xzvf source.tar.gz = extracts / unpacks files from compressed file

**History**

history = shows history of commands I have done

SHELL SCRIPTING

Shell = primary program that computers use to receive code (i.e. commands)

Bash (= Bourne Again Shell) = implementation of Shell – allos you directly, efficiently performmany tasks

Git Bash = an application that provides Git command line experience on operating system

I am using Git Bash / Command language

.sh = extension for a bourne shell script

chmod +x myscript.sh = make a script executable

./myscript.sh = runs a script

which bash = shows me, where the bash is located

git init = initiates git command – creates a hidden directory .git/ (all history of the project)

(master) = master branch

git help

usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]

[--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]

[-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]

[--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]

[--config-env=<name>=<envvar>] <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)

clone Clone a repository into a new directory

init Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)

add Add file contents to the index

mv Move or rename a file, a directory, or a symlink

restore Restore working tree files

rm Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)

bisect Use binary search to find the commit that introduced a bug

diff Show changes between commits, commit and working tree, etc

grep Print lines matching a pattern

log Show commit logs

show Show various types of objects

status Show the working tree status

grow, mark and tweak your common history

branch List, create, or delete branches

commit Record changes to the repository

merge Join two or more development histories together

rebase Reapply commits on top of another base tip

reset Reset current HEAD to the specified state

switch Switch branches

tag Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)

fetch Download objects and refs from another repository

pull Fetch from and integrate with another repository or a local branch

push Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some

concept guides. See 'git help <command>' or 'git help <concept>'

to read about a specific subcommand or concept.

See 'git help git' for an overview of the system.

1. vytvořím file např. touch demo.txt
2. přidám file do „stage area“ add demo.txt
3. dám git status a vidím, co má být uděláno (changes to be committed)
4. dám commit a tím se file někam? dostane

touch CREATE

add STAGE (what exactly is staging??)

commit COMMIT

Paralelně s tím: GitHub (Neconoveho\_2024!)

Bash repository = local, only on my machine

Git Basic Commands

gitmygit init // initialize local Git repository

git add <file> // add file(s) to index

git status // check status of working tree

git commit // comit changes to index (gets everything that is in staging area and puts in to the local repository)

git push // take the local repository and pushe it to the remotes repository (GitHub)

git pull // pull latest from remote repository

git clone // copy remote repository into a current / new directory