

# Introduction to OnDemand

Todd Fallesen

Slides by Vanessa Dao, Chris Hadjigeorgiou

The top part of the image shows the Nemo desktop interface. It has a green header bar with the text "Nemo desktop (2192913)" and "1 node | 6 cores | Running". Below this, there's a "Host:" field with the value ">\_gpu025.camp.thecrick.org" and a "Delete" button. The "Created at:" field shows "2024-04-22 09:01:00 BST". The "Time Remaining:" field shows "5 hours and 39 minutes". The "Session ID:" is "9c5f19fd-aecf-4d01-91ee-b8df23c7cd9c". There are two sliders: "Compression" (0 (low) to 9 (high)) and "Image Quality" (0 (low) to 9 (high)). At the bottom of this section are two buttons: "Launch Nemo desktop" and "View Only (Share-able Link)".

The bottom part of the image shows a terminal window with the prompt "daov1@gpu025:~". The user has entered the command "conda env list". The output shows a list of conda environments and their paths:

```
(base) [daov1@gpu025 ~]$ conda env list
# conda environments:
#
base                * /camp/apps/eb/software/Anaconda3/2023.09-0
StarDist_Crick_Feb24 /camp/home/daov1/.conda/envs/StarDist_Crick_Feb24
brainglobe          /camp/home/daov1/.conda/envs/brainglobe
cellpose_attempt     /camp/home/daov1/.conda/envs/cellpose_attempt
cp426_nemo_v1        /camp/home/daov1/.conda/envs/cp426_nemo_v1
csbdeep-env          /camp/home/daov1/.conda/envs/csbdeep-env
napari-env           /camp/home/daov1/.conda/envs/napari-env
napari-env1-nemo     /camp/home/daov1/.conda/envs/napari-env1-nemo
napari-env4-nemo     /camp/home/daov1/.conda/envs/napari-env4-nemo
napari-nemo-april24  /camp/home/daov1/.conda/envs/napari-nemo-april24
napari-nemo-feb24    /camp/home/daov1/.conda/envs/napari-nemo-feb24
napari-nemo-jan24    /camp/home/daov1/.conda/envs/napari-nemo-jan24
napari-plugins       /camp/home/daov1/.conda/envs/napari-plugins
napari-tracking      /camp/home/daov1/.conda/envs/napari-tracking
pyclesperanto_env    /camp/home/daov1/.conda/envs/pyclesperanto_env
ultrack-attempt      /camp/home/daov1/.conda/envs/ultrack-attempt
ultrack-flow-field   /camp/home/daov1/.conda/envs/ultrack-flow-field

(base) [daov1@gpu025 ~]$
```

# Quick Overview

## NEMO

- What even is it
- How to start jobs

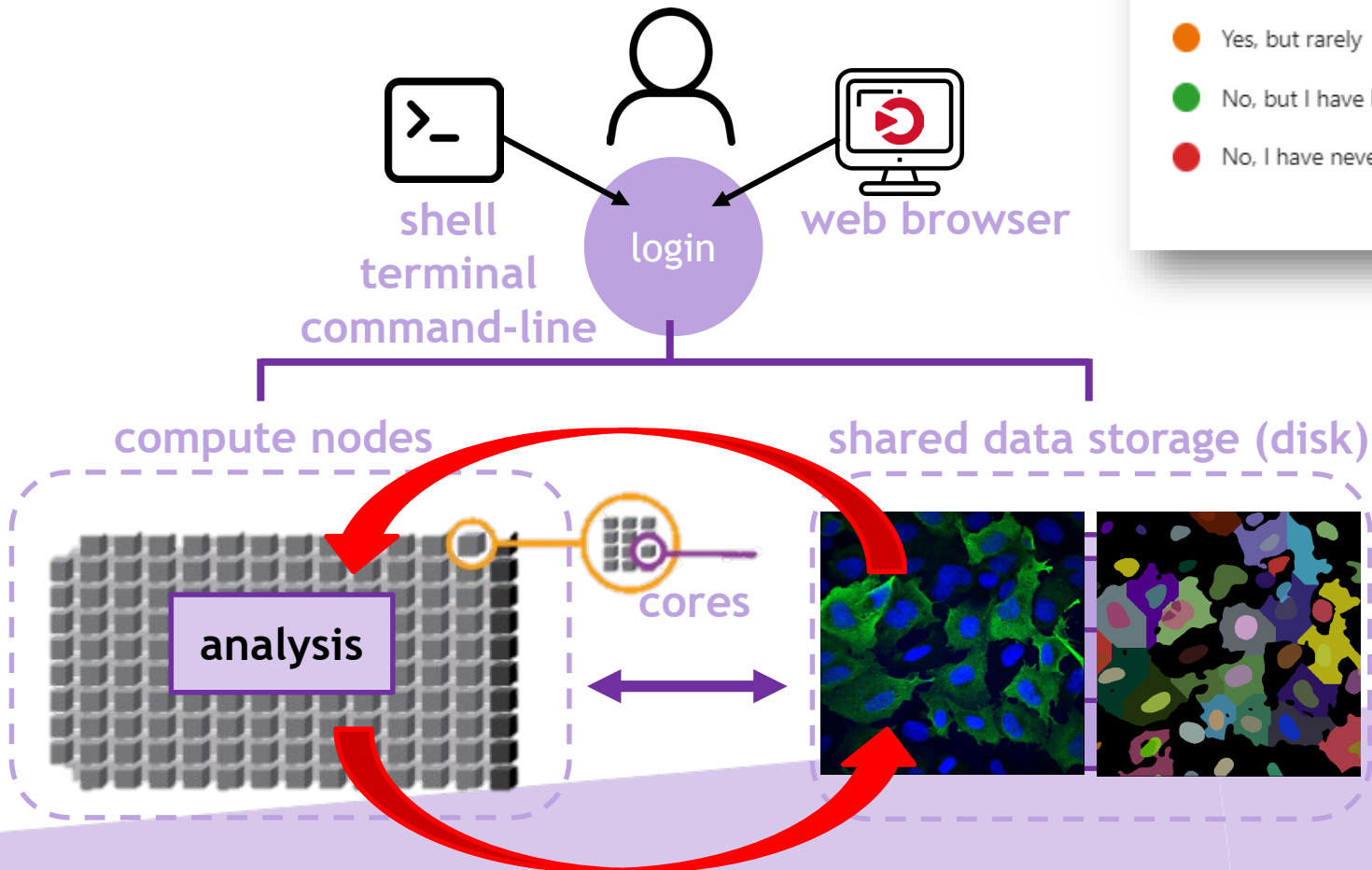
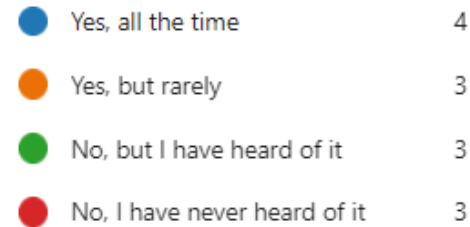
## OnDemand desktop

- Why it's great (when not broken)
- How to navigate the terminal
- Modules
- Installing and Launching a Program (FIJI)

## Anaconda

- Set up Anaconda on OnDemand
- Link Jupyter Notebook To Conda Environment

- It is BOTH storage and the computing system

[More Details](#)

# Nemo Overview

**Nemo is currently composed of:**

- **5 partitions which will be publicly available soon(AMD based):**
  - ncpu: 92 CPU nodes, 256 threads, 2TB RAM
  - gpu : 40 GPU nodes, 80 threads, 754GB RAM, 4x NVidia V100 GPGPU(32GB RAM)
  - nhmem: 2 High Memory, 256 threads, 4TB RAM
  - nint: 4 Interactive nodes, 256 threads, 2TB RAM
  - vis : 4 Visualization nodes, 48 threads, 376GB RAM, Quadro RTX 5000, only available with srun/salloc

**Any user can submit jobs to any queue but each queue has different constraints.**

- **Storage**
  - 16+ PB GPFS Filesystem
- **Networking to each compute and storage**
  - [200/100/56]Gb/s (NDR/HDR/FDR) Infiniband network
  - 40Gb/s Ethernet network

- **OnDemand**

- In your browser go to <https://ondemand.nemo.thecrick.org>

- **SSH**

- For Windows download and install MobaXterm, Powershell or PuTTY. Start a new SSH session and set login.nemo.thecrick.org as host, your Crick username
- For Mac/Linux simply open a terminal and use the following:  
`ssh -i $HOME/.ssh/id_rsa username@login.nemo.thecrick.org`

## **Important:**

**Never share any credentials or the private part of an SSH key. Try to use different keys for different systems and always password protect them. Use SSH agent and password manager for convenience.**

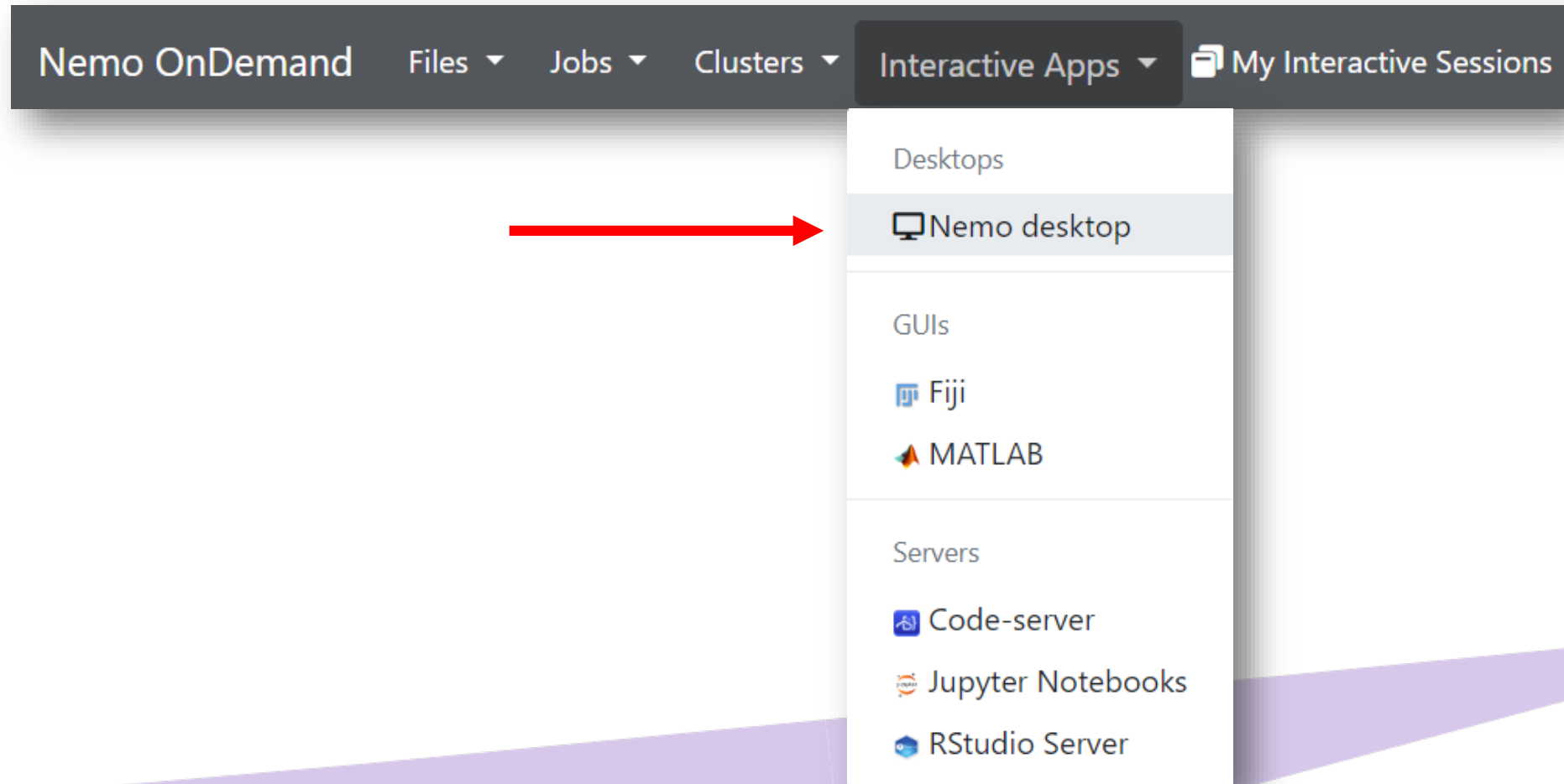
# OnDemand overview and basics



- OnDemand is a web portal for access to HPC resources on CAMP
- Provides:
  - Shell access
  - File manager - currently limited to personal home directory
  - Interactive apps
  - Queue listing/job management
- The interface has two main components:
  - The main window
  - The dashboard

# NEMO OnDemand

<https://ondemand.nemo.thecrick.org/>



# Starting jobs on NEMO

## Nemo desktop

This app will launch an interactive desktop on one or more compute nodes. You will have full access to the resources these nodes provide. This is analogous to an interactive batch job.

Partition

gpu

Please select a partition from the drop-down menu

Partition	Nodes	Cores per node	Memory per node
gpu	40	80	750GB
ncpu	96	256	2000G
nhmem	2	256	4000G

Number of hours

1

Number of nodes

1

Number of tasks

1

Number of cores/tasks per node.

Memory per node

1

2 Start X server(only on GPU node)

Select this to start an X server for graphics acceleration if using a gpu node(experimental).

Use VirtualGL to start an application with acceleration:

```
m1 VirtualGL
vglrun application
```

Reservation

ondemand

To submit the job to a reservation enter the reservation name here, otherwise leave blank.

Launch



# Starting jobs on NEMO

Job ID

Nemo desktop (2424965) 1 node | 6 cores | Running

Host: [\\_gpu020.camp.thecrick.org](https://_gpu020.camp.thecrick.org) Delete

Created at: 2024-04-24 09:01:44 BST

Time Remaining: 16 hours and 41 minutes ← Time until job **DIES**

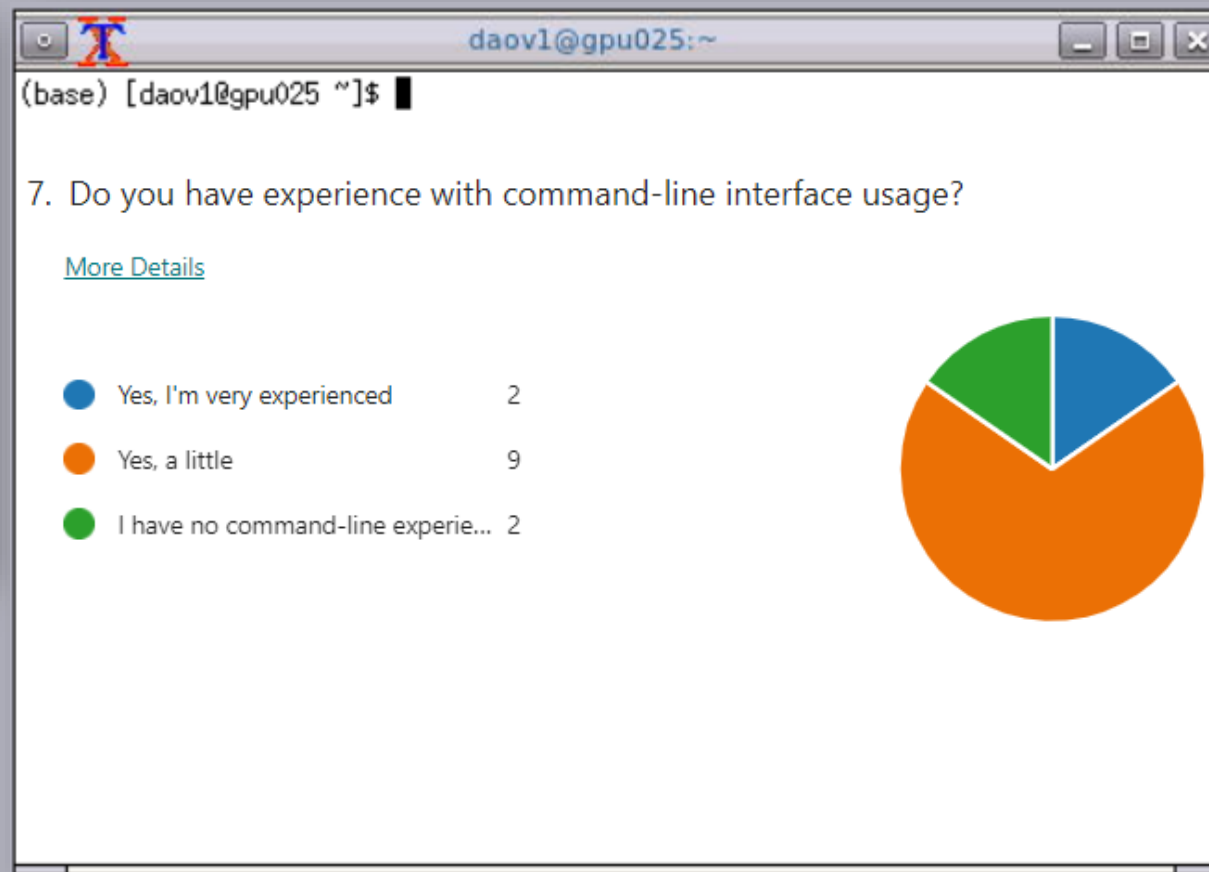
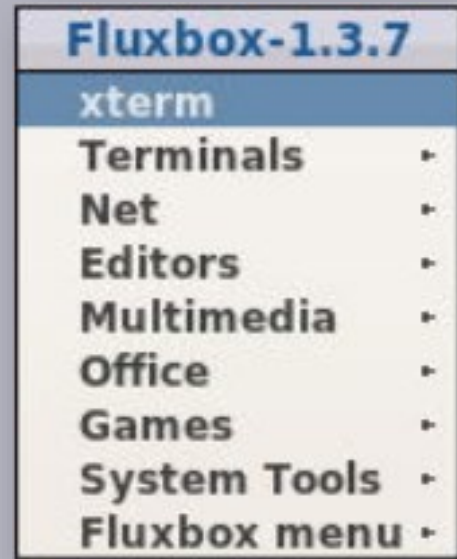
Session ID: 36f396aa-7c50-4307-b84c-4e0041718aff

Compression 0 (low) to 9 (high) Image Quality 0 (low) to 9 (high)

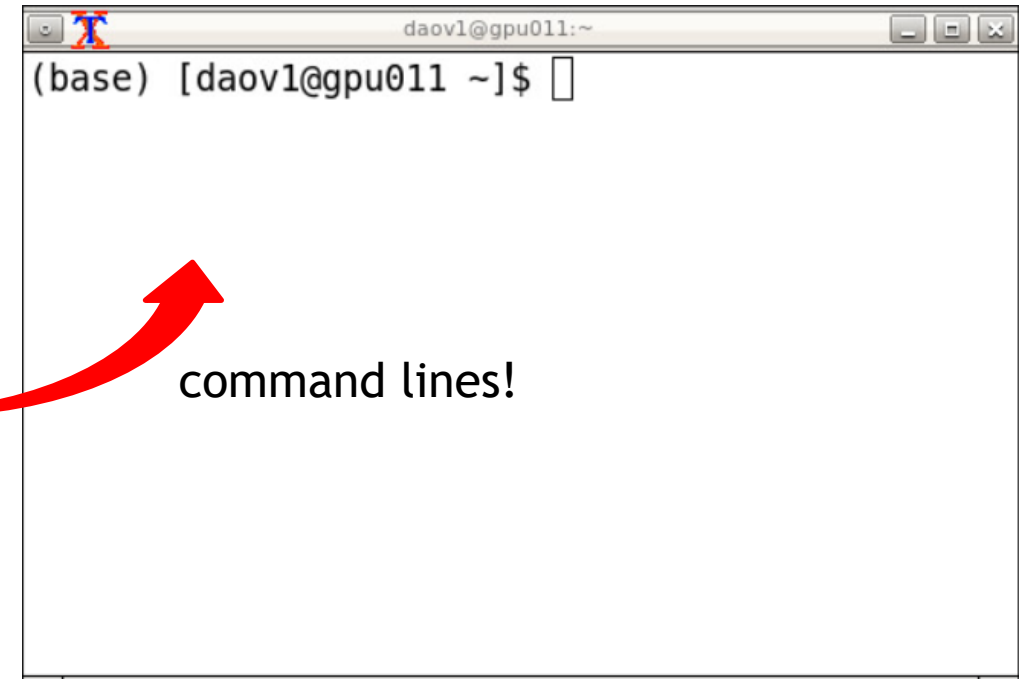
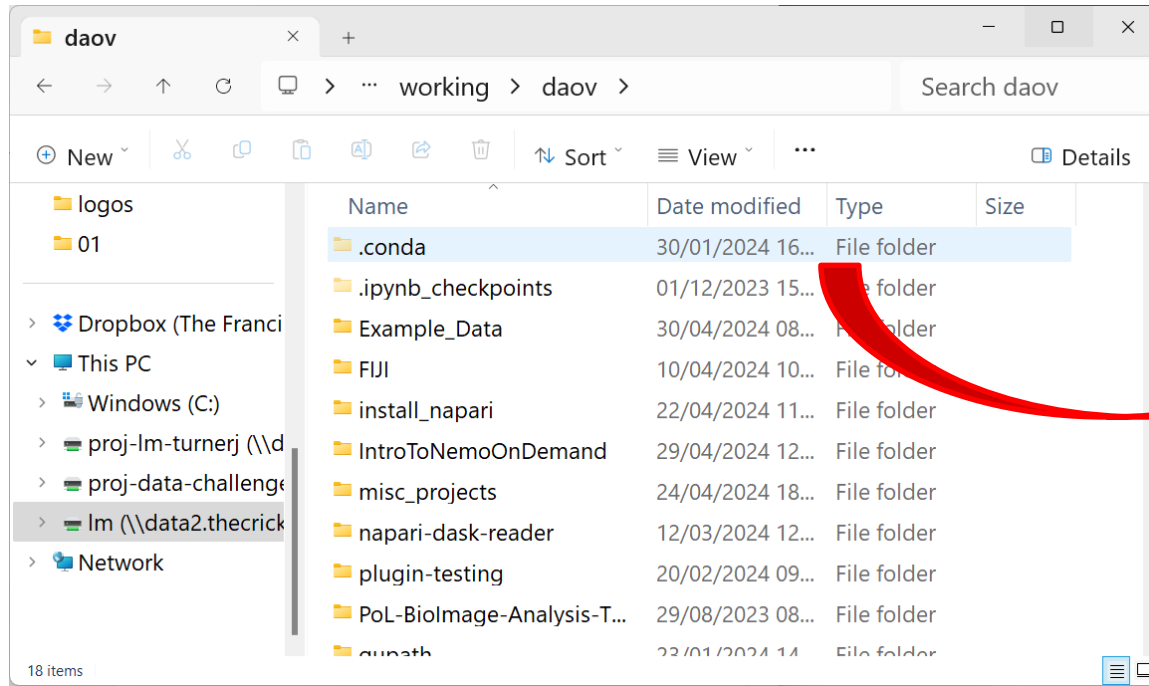
Launch Nemo desktop View Only (Share-able Link)

Click to start

# Mastering the VNC



# REMEMBER: Nemo is storage too!

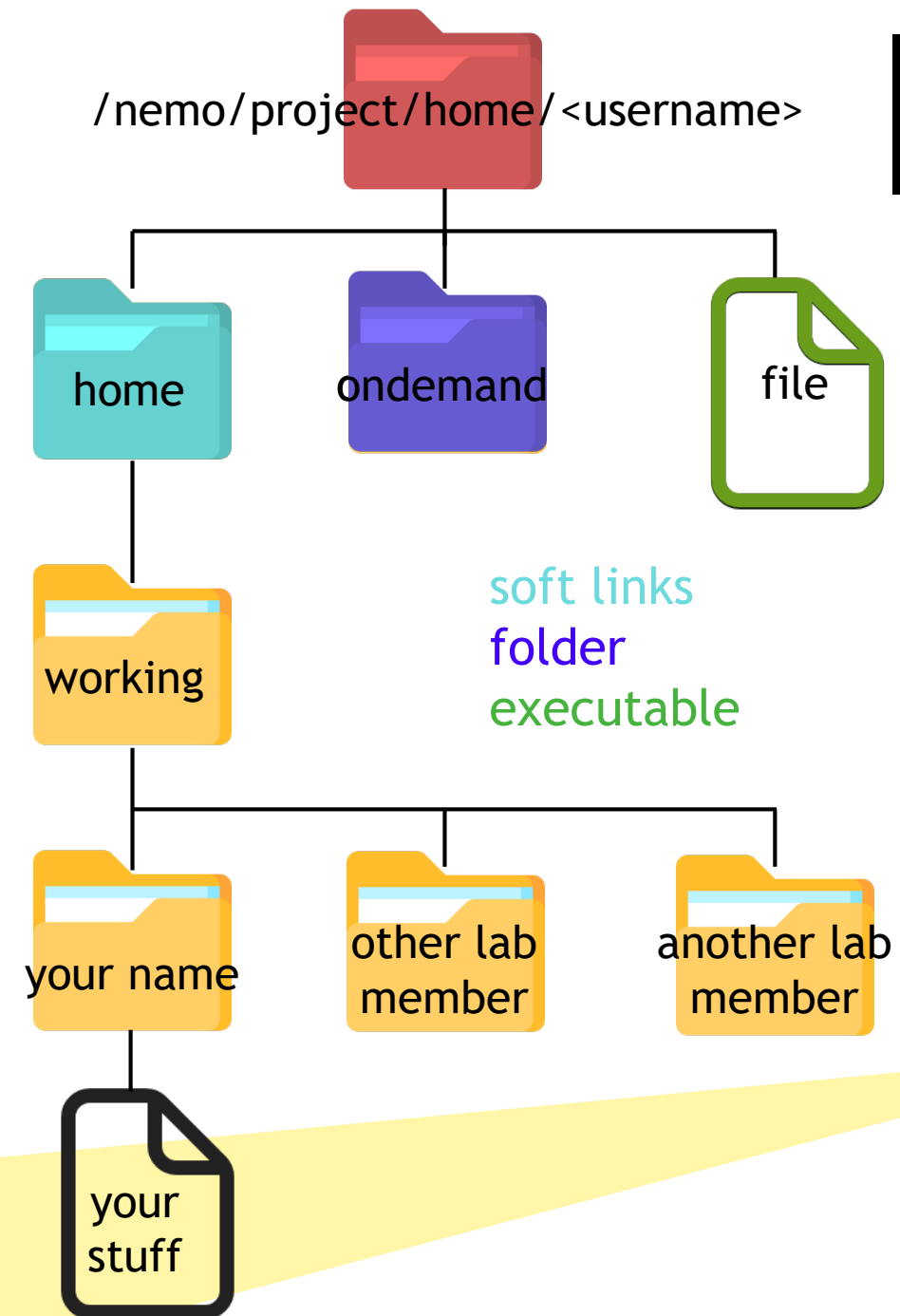


command lines!

# Mastering the terminal

```
daov1@gpu011 ~/home
(base) [daov1@gpu011 ~]$ realpath .
/nemo/project/home/daov1
(base) [daov1@gpu011 ~]$ ls
CellProfilerLocal.cfg
config
daov
dummy
home
ondemand
Run_QuPath.sh
<%= session.staged_root.join(fluxbox.rc) %>
slurm-55063720.out
(base) [daov1@gpu011 ~]$ cd home
(base) [daov1@gpu011 home]$ ls
archive  inputs  outputs  reference  scratch  working  www
(base) [daov1@gpu011 home]$
```

```
realpath .
ls
ls -a
cd path/to/directory
cd ..
cd $HOME
```



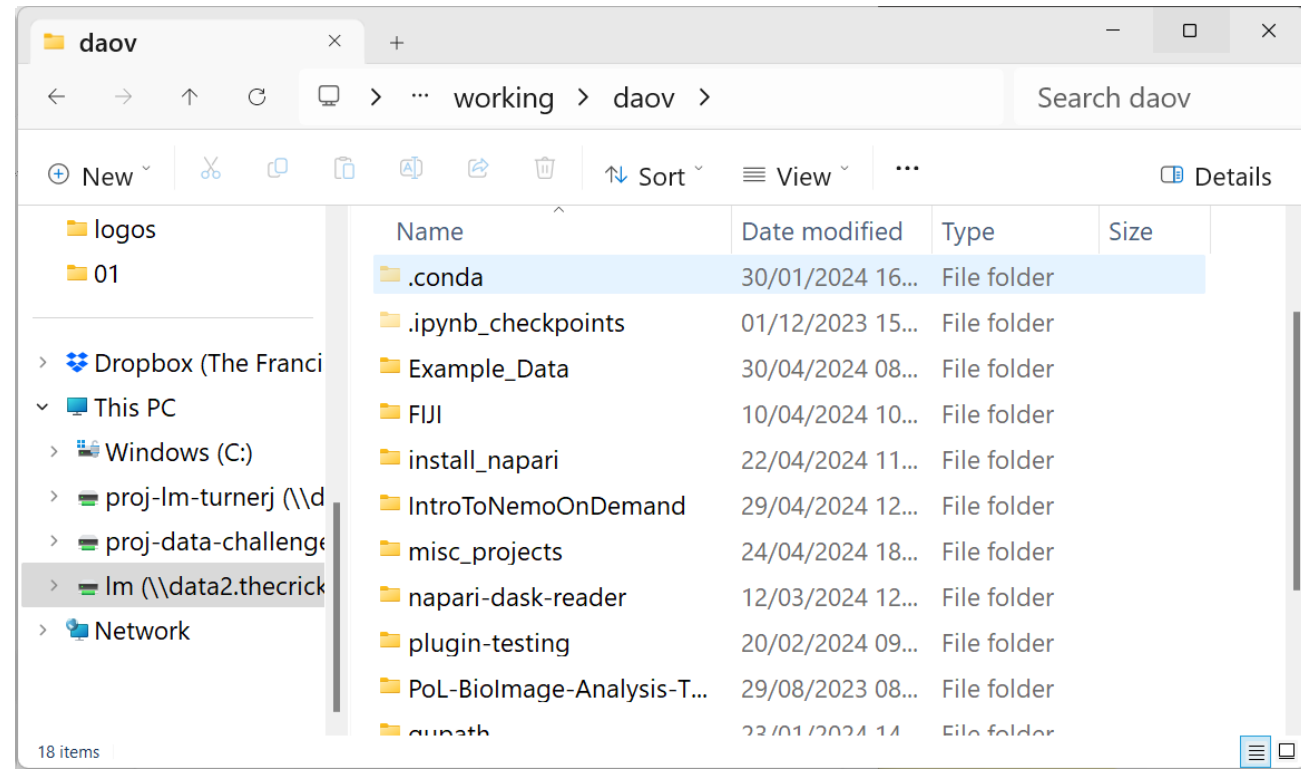
# Command line basics

- **Get directory listing**
  - *ls*
  - *ls path*
- **Print working directory**
  - *pwd*
  - *realpath .*
- **Change directory**
  - *cd path*
  - *cd #* without arguments takes you back to your home directory
  - *cd .. #* change to the directory one level up

# Getting to know the terminal

```
daov1@gpu011:~/home/working/daov
(base) [daov1@gpu011 home]$ cd working
(base) [daov1@gpu011 working]$ ls
admin          daov          Mashanov
amazul         DataPurge copy.txt Recruitment
andersk        DataPurge.README rensam
archived_folders Demo systems  shared
aubynd         fallest      spacefm.sh
barryd         folder_sizes.txt spitzers
belld          greenwh      Training
charoyc        guntond      User Info
ciccara        hok
dantuor        ilastik.sh
(base) [daov1@gpu011 working]$ cd daov
(base) [daov1@gpu011 daov]$ ls
Example_Data
FIJI
install_napari
misc_projects
napari-dask-reader
plugin-testing
PoL-BioImage-Analysis-TS-GPU-Accelerated-Image-Analysis-main
```

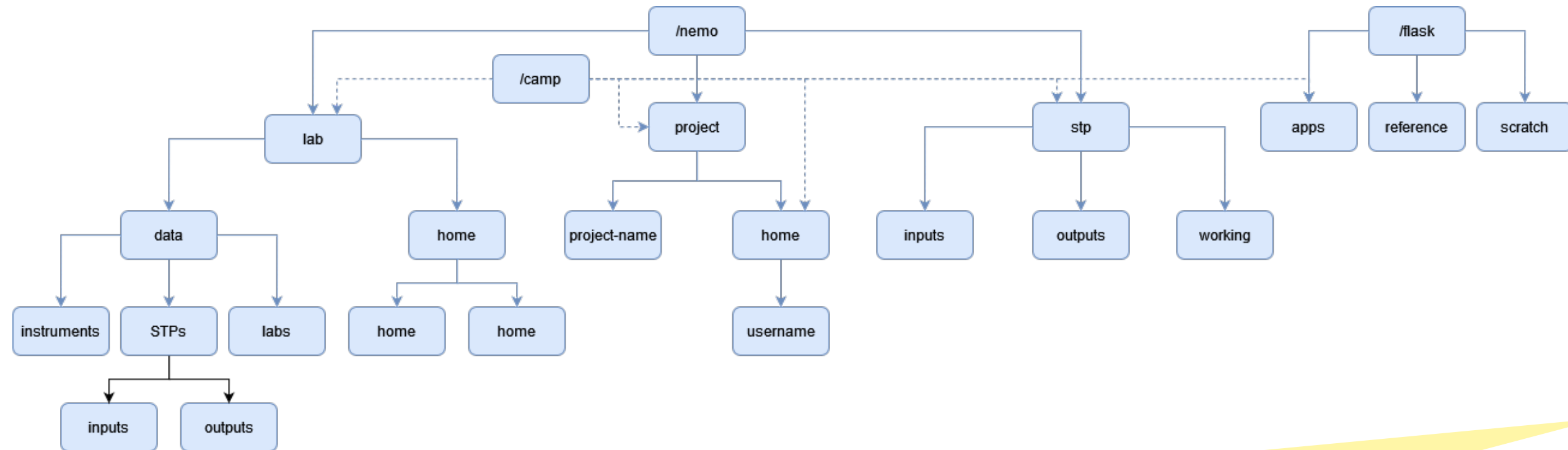
```
realpath .
ls
ls -a
cd path/to/directory
cd ..
cd $HOME
```



```
(base) [daov1@gpu011 daov]$ realpath .
/nemo/stp/lm/working/daov
(base) [daov1@gpu011 daov]$
```

Find the path to your folder in your lab space

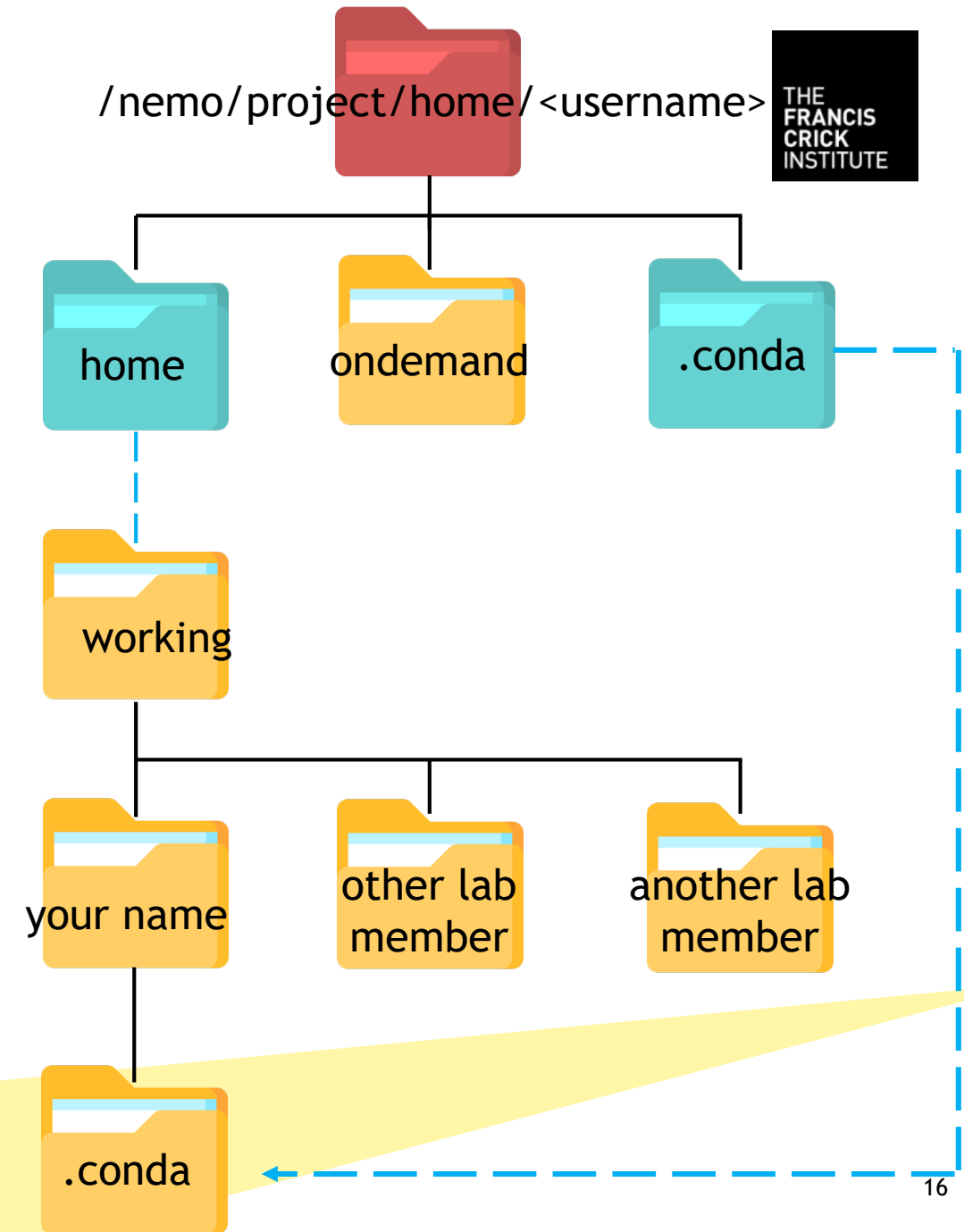
# NEMO's folder structure



# Soft linking

```
(base) [daov1@gpu004 ~]$ ls -a -l
total 168
drwx----- 32 daov1 domain users 16384 Apr 25 09:53 .
drwxr-xr-x 1505 root root 65536 Apr 25 09:57 ..
-rw----- 1 daov1 domain users 26010 Apr 25 02:13 .bash_history
-rw-r--r-- 1 daov1 domain users 18 Jul 7 2023 .bash_logout
-rw-r--r-- 1 daov1 domain users 141 Jul 7 2023 .bash_profile
-rw-r--r-- 1 daov1 domain users 935 Jan 26 11:27 .bashrc
lrwxrwxrwx 1 daov1 domain users 16 Feb 21 13:42 .brainglobe -> daov/.brainglobe
drwxr-xr-x 2 daov1 domain users 4096 Mar 11 15:20 .brainrender
lrwxrwxrwx 1 daov1 domain users 11 Jan 25 12:06 .cache -> daov/.cache
lrwxrwxrwx 1 daov1 domain users 14 Feb 28 13:38 .cellpose -> daov/.cellpose
-rw-r--r-- 1 daov1 domain users 53 Dec 5 14:05 CellProfilerLocal.cfg
lrwxrwxrwx 1 daov1 domain users 41 Jul 30 2023 .conda -> /camp/home/daov1/home/working/daov/.conda
-rw-r--r-- 1 daov1 domain users 17 Apr 22 11:51 .condarc
-rw-r--r-- 1 daov1 domain users 144 Sep 28 2023 config
drwxr-xr-x 11 daov1 domain users 4096 Apr 22 10:23 .config
drwxr-xr-x 2 daov1 domain users 4096 Jan 2 14:41 .cookiecutter_replay
drwxr-xr-x 3 daov1 domain users 4096 Jan 2 14:40 .cookiecutters
drwxr-xr-x 3 daov1 domain users 4096 Aug 22 2023 .cupy
lrwxrwxrwx 1 daov1 domain users 34 Jul 28 2023 daov -> /camp/home/daov1/home/working/daov
```

```
realpath .
ls
ls -a
ls -a -l
cd path/to/directory
cd ..
cd $HOME
```





# Software Modules

- Modules are centrally installed software on Nemo available to all users
- Accessible through the module(or ml) command
- To check if a module is available either of the following can be used:
  - *ml avail software*
  - *ml spider software*
- To load a module:
  - `ml modulename`
- List currently loaded modules
  - `ml`

# Software Modules



- To remove a module
  - *ml unload modulename*
- To remove all loaded modules
  - *ml purge*
- You can also create your own modules or use modules shared by a colleague or different lab/STP. To do so you just need to add the path of the directory which contains those modules:
  - *ml use /path/to/directory*

## Exercise 1 (ml)

- List the SpaceFM module - ml av SpaceFM
- Load SpaceFM - ml SpaceFM
- Run SpaceFM(Case might be different) - spacefm
- Check what modules were loaded - ml
- Investigate Nemo's folder structure
- Close SpaceFM
- Unload the SpaceFM module - ml unload SpaceFM
- Check if any modules are still loaded - ml
- Purge all the modules - ml purge

Tip: Some commands can autocomplete using the Tab key, ml can do that with module names

# Command line basics

- Copy files or directories
  - *cp source destination*
  - *cp -R source\_directory destination\_directory*
- Move files/directories
  - *mv source destination*
- Create a new directory
  - *mkdir /path/to/directory* # Assumes parent directories already exist
  - *mkdir -p /path/to/new/directory* # Creates parent directories if they don't exist
- Print or display a file
  - *cat filename* # prints file contents, great for small files
  - *less filename* # display the files in an interactive viewer, *q* to quit
- Manual pages
  - *man ls*

# Fiji

- A distribution of ImageJ which includes many useful plugins
- You'll setup a personal Fiji installation for easier plugin management
- Start a Remote Desktop session
- Open xterm(right click on desktop to open menu)
- Copy the fiji app from the training folder to your user folder in your lab
- Unzip the file
  - Unzip fiji\_linux64.zip
- Go into the directory which was extracted
  - cd Fiji.app
- Launch fiji with ./ImageJ-linux64
- <https://downloads.imagej.net/fiji/latest/fiji-linux64.zip>

## Exercise 2 (pwd, ls, cd, cp, df, cat, less, man)

- Open a terminal/shell and check which directory you're currently in
  - `realpath .`
- List the files in the directory
  - `ls`
- Optional - List the hidden files in the directory
  - `ls -a`
- Go to your lab home or working directory
  - `cd /nemo/lab/<labname>/home/users/<username>`
- Go back to your own home and copy the fiji file  
*/camp/apps/training/stardist/StarDist\_Course\_Sept\_2024/fiji-linux64.zip to your home*
  - `cp /camp/apps/training/stardist/StarDist_Course_Sept_2024/fiji-linux64.zip .`
- Check how much space you have available in your personal home
  - `df -h $HOME`
- Unzip FIJI

# Launch Fiji in OnDemand (Exercise 3)



- **We will write a shell script to do this**
  - File should be in home directory
  - Need to change directory to where you saved the Fiji installation
  - Need to launch the Fiji app
- **Use nano, or notepad ++**
  - nano is a quick and easy terminal based text editor
  - In home directory, type in “nano”
  - Type in your commands
  - Cntrl-x will exit, and prompt you to save
  - Enter “Run\_Fiji.sh” as the file name
  - `chmod u+rx Run_Fiji.sh`

napari-environment



Python 3.9

cellprofiler-environment



Python 3.8

stardist-environment



Python 3.10

## Initialising Anaconda

```
daov1@gpu020:~  
(base) [daov1@gpu020 ~]$  
  
ml Anaconda3/2023.09-0  
conda init bash  
ml purge  
  
conda env list
```

Don't install anything in your base environment!

Always activate your environment before installing packages!

Package and environment management software

Conda environments are a method for creating an isolated space with different software and their dependencies installed.

Conda environments can also help with reproducibility.



# Your turn: Soft linking

1. Check if you already have **.conda** and **.cache** in your **\$HOME** directory

```
cd $HOME  
ls -a
```

2. If **softlinked .conda** and **.cache** exists then **DONE**

2. If folder **.conda** and **.cache** exists then move to your lab space

```
mv .conda /path/to/lab/space  
mv .cache /path/to/lab/space
```

2. If they don't exist then create in your lab space

```
mkdir /path/to/lab/space/.conda  
mkdir /path/to/lab/space/.cache
```

```
realpath .  
ls  
ls -a  
ls -a -l  
cd path/to/directory  
cd ..  
cd $HOME
```

3. Create the softlink in your **\$HOME** directory

```
ln -s /path/to/lab/space/.conda .  
ln -s /path/to/lab/space/.cache .
```

# Conda initialisation

- .conda is the folder where Conda stores environments and packages by default. Conda automatically creates that directory the first time it's run in your home directory. Since users' home directories have limited space we create(or move) .conda in their lab folder and use a symbolic link from the personal home directory.
- \$HOME is an environment variable with the path of your home directory.
- Now you can initialize Conda:
  - *conda init*

# Conda with Jupyter in OnDemand



- To integrate conda environments with Jupyter in OnDemand we'll make use of remote kernels. This only needs to be done once for each environment.
- First install the necessary packages inside your conda environment:
  - `conda activate testenv`
  - `pip install remote-ikernel ipykernel`
- Install an ipython kernel (necessary for Jupyter):
  - `python -m ipykernel install --user --name=testenv`
- Install the remote Jupyter kernel:
  - `python3 -m remote_ikernel manage --add --kernel_cmd="ml purge && conda activate testenv && ipython3 kernel -f {connection_file}" --name="testenv" --interface=local --workdir=~/" --language=python3`
- You should be able to launch a kernel now with your conda environment called "Local testenv"

# Understanding the commands

- Install the remote Jupyter kernel:
  - `python3 -m remote_ikernel manage --add --kernel_cmd="ml purge && conda activate testenv && ipython3 kernel -f {connection_file}" --name="testenv" --interface=local --workdir="~/ " --language=python3`
  - **Text in red:** launching the python program remote\_ikernel manage
  - **Text in green:** The commands were adding to each time Jupyter is launched from this environment
  - **Text in purple:** The environment name we are adding to Jupyter notebooks
  - Remote\_ikernel manager is the program in python which links environments into Jupyter notebooks, or else they don't show in jupyter

# SLURM

- Most HPC systems utilise job schedulers to manage how users can access resources.
- Nemo uses Slurm for job scheduling on the compute resources.
- A number of policies/limits are implemented to allow fair usage by labs/users

# SLURM commands

- Check your jobs in the queue  
***squeue -u username***
- Get overall info for all the partitions  
***sinfo***
- Submit a batch script/job(only for cpu/gpu/hmem partitions)  
***sbatch script.sh***
- Start an interactive job  
***srun -p int -t 00:10:00 --pty bash***

## Exercise 4

- Copy the sbatch-example-new.sh from the training directory to your home folder
  - `cp /camp/apps/training/stardist/sbatch-example-new.sh $HOME/`
- Modify it with following (you can edit the file via the OnDemand File explorer):
  - Change the partition from ncpu to gpu
    - `#SBATCH --partition=ncpu`
  - Comment out memory per cpu (add another # to line beginning)
    - `##SBATCH --mem-per-cpu=`
- Submit the job with sbatch
  - `sbatch sbatch-example-new.sh`
- Check output with *cat* or *less* command
  - `cat slurm-<jobid>.out`