

Scientific Computing @ CLASSE

The CLASSE Linux System

2025-06-06

Overview

- CLASSE Linux computers
- How to log in
- Where to work on your projects
- Practice using the terminal

The CLASSE Linux System

Where to work – computers

- `lnx201.classe.cornell.edu`
 - General purpose login node – a *shared* resource
 - Use for file browsing, text editing, running code that doesn't consume many resources
- CLASSE compute farm
 - <https://wiki.classe.cornell.edu/Computing/ComputeFarmIntro>
 - Use for running code that may consume a lot of resources
- Station computers: `idxx.classe.cornell.edu`
 - Connected to station hardware (beamstops, motors, detectors, etc.)
 - Use ONLY if you are controlling station hardware
- More guidance on where to work: <https://wiki.classe.cornell.edu/Computing/WhichComputer>

The CLASSE Linux System

Logging in

ssh — secure shell

- On Linux & mac terminals:

```
ssh <CLASSEID>@<host>
```

- For example:

```
ssh kls286@lnx201.classe.cornell.edu
```

On Windows: PuTTY

- <https://wiki.classe.cornell.edu/Computing/WinTunnelVncSSH>

NoMachine — full desktop interface

- On any computer, open an web browser and visit:

<https://nomachine.classe.cornell.edu>

- Be prepared for issues with speed, copy/paste, and dropped connections

Another CLASSE-specific option:

<https://jupyterhub.classe.cornell.edu>

The CLASSE Linux System

Exercise 1

Open a terminal on `lnx201` using one of the following options:

1. `ssh` from your computer's terminal (for Mac and Linux users)
2. PuTTY (for Windows users)
3. <https://nomachine.classe.cornell.edu>
4. <https://jupyterhub.classe.cornell.edu>

The CLASSE Linux System

Exercise 2

Start navigating in the same terminal as before. Run the following commands:

1. `pwd`

- “Print working directory” tells you what the current working directory is

2. `ls`

- Lists the contents of the current working directory

3. `ls -la`

- Lists the contents of the current working directory with the additional options:
 - `-l` tells `ls` to show details about each file’s type, permissions, size, etc. (“l” for “long”)
 - `-a` tells `ls` to list hidden files, too (“a” for “all”)
- Individual options to `ls` like `-l` and `-a` can be shortened to `-la`

4. `cd <directory of your choice>`

- “change directory” changes your current working directory to the specified destination

5. `cd`

- `cd` with no arguments takes you back to your home directory

The CLASSE Linux System

Where to work – directories

- [HomeDisk](#) (/home/<CLASSE-ID>) is limited to 1GB
 - This is the landing point when you open a new terminal or log in with ssh
 - **Do not make a habit of working in your home directory!**
- Recommended working space: /nfs/...
 - For CHESS students: /nfs/chess/user/<CLASSE-ID>/
 - For other students: ask your project mentor
 - For these exercises, use /cdat/tem/<CLASSE-ID>/
 - `nfs` stands for Network File System – if you put something in /nfs on `lnx201`, it will also be there on the CLASSE Compute Farm nodes, the station computers, the computers in the CESR and CHESS operations areas, etc.

The CLASSE Linux System

Exercise 3

Make a symbolic link to your `/cdat/tem/<your CLASSE ID>/` directory inside your home directory.

1. In the terminal you opened before, run this command (make sure to substitute appropriate values where something is enclosed in `<>` before running):

```
ln -s /cdat/tem/<your CLASSE ID>/ ~/<your link name>
```

The CLASSE Linux System

Getting comfortable in the terminal

- A fantastic intro to the Linux command line in general and at CLASSE in particular:
<https://xcitecourse.org/theme2/SF100/>
 - Thanks to our collaborators from X-CITE (CyberInfrastructure Training and Education for Synchrotron X-Ray Science)!
- A nice cheat sheet of Linux commands —>
- If you don't know how to use a command, try running one of the following to get a help menu / manual entry for it:

- `<command> -h`
- `<command> -help`
- `man <command>`

Command	Description
<code>pwd</code>	prints working directory (prints to screen, ie displays the full path, or your location on the filesystem)
<code>ls</code>	lists contents of current directory
<code>ls -l</code>	lists contents of current directory with extra details
<code>ls /home/user/*.txt</code>	lists all files in /home/user ending in .txt
<code>cd</code>	change directory to your home directory
<code>cd ~</code>	change directory to your home directory
<code>cd /scratch/user</code>	change directory to user on scratch
<code>cd -</code>	change directory to the last directory you were in before changing to wherever you are now
<code>mkdir mydir</code>	makes a directory called mydir
<code>rmdir mydir</code>	removes directory called mydir. mydir must be empty
<code>touch myfile</code>	creates a file called myfile. updates the timestamp on the file if it already exists, without modifying its contents
<code>cp myfile myfile2</code>	copies myfile to myfile2. if myfile2 exists, this will overwrite it!
<code>rm myfile</code>	removes file called myfile
<code>rm -f myfile</code>	removes myfile without asking you for confirmation. useful if using wildcards to remove files ***
<code>cp -r dir newdir</code>	copies the whole directory dir to newdir. -r must be specified to copy directory contents recursively
<code>rm -rf mydir</code>	this will delete directory mydir along with all its content without asking you for confirmation! ***
<code>nano</code>	opens a text editor. see ribbon at bottom for help. ^x means CTRL-x. this will exit nano
<code>nano new.txt</code>	opens nano editing a file called new.txt
<code>cat new.txt</code>	displays the contents of new.txt
<code>more new.txt</code>	displays the contents of new.txt screen by screen. spacebar to pagedown, q to quit
<code>head new.txt</code>	displays first 10 lines of new.txt
<code>tail new.txt</code>	displays last 10 lines of new.txt
<code>tail -f new.txt</code>	displays the contents of a file as it grows, starting with the last 10 lines. ctrl-c to quit.
<code>mv myfile newlocdir</code>	moves myfile into the destination directory newlocdir
<code>mv myfile newname</code>	renames file to newname. if a file called newname exists, this will overwrite it!
<code>mv dir subdir</code>	moves the directory called dir to the directory called subdir
<code>mv dir newdirname</code>	renames directory dir to newdirname
<code>top</code>	displays all the processes running on the machine, and shows available resources
<code>du -h --max-depth=1</code>	run this in your home directory to see how much space you are using. don't exceed 5GB
<code>ssh servername</code>	goes to a different server. this could be queso, brie, or provolone
<code>grep pattern files</code>	searches for the pattern in files, and displays lines in those files matching the pattern
<code>date</code>	shows the current date and time
<code>anycommand > myfile</code>	redirects the output of anycommand writing it to a file called myfile
<code>date > timestamp</code>	redirects the output of the date command to a file in the current directory called timestamp
<code>anycommand >> myfile</code>	appends the output of anycommand to a file called myfile
<code>date >> timestamp</code>	appends the current time and date to a file called timestamp. creates the file if it doesn't exist
<code>command1 command2</code>	"pipes" the output of command1 to command2. the pipe is usually shift-backslash key
<code>date grep Tue</code>	displays any line in the output of the date command that matches the pattern Tue. (is it Tuesday?)
<code>tar -zxf archive.tgz</code>	this will extract the contents of the archive called archive.tgz. kind of like unzipping a zipfile. ***
<code>tar -zcf dir.tgz dir</code>	this creates a compressed archive called dir.tgz that contains all the files and directory structure of dir
<code>time anycommand</code>	runs anycommand, timing how long it takes, and displays that time to the screen after completing anycommand
<code>man anycommand</code>	gives you help on anycommand
<code>cal -y</code>	free calendar, courtesy unix
<code>CTRL-c</code>	kills whatever process you're currently doing
<code>CTRL-insert</code>	copies selected text to the windows clipboard (n.b. see above, ctrl-c will kill whatever you're doing)
<code>SHIFT-insert</code>	pastest clipboard contents to terminal

*** = use with extreme caution! you can easily delete or overwrite important files with these.

Absolute vs relative paths.

Let's say you are here: /home/turnersd/scripts/. If you wanted to go to /home/turnersd/, you could type: `cd /home/turnersd/`. Or you could use a relative path. `cd ..` (two periods) will take you one directory "up" to the parent directory of the current directory.

`.` (a single period) means the current directory
`..` (two periods) means the parent directory
`~` means your home directory

A few examples

<code>mv myfile ..</code>	moves myfile to the parent directory
<code>cp myfile ../newname</code>	copies myfile to the parent directory and names the copy newname
<code>cp /home/turnersd/scripts/bstrap.pl .</code>	copies bstrap.pl to "." i.e. to dot, or the current directory you're in
<code>cp myfile ~/subdir/newname</code>	copies myfile to subdir in your home, naming the copy newname
<code>more ../../../../myfile</code>	displays screen by screen the content of myfile, which exists 3 directories "up"

Wildcards (use carefully, especially with rm)

`*` matches any character. example: `ls *.pl` lists any file ending with ".pl"; `rm dataset*` will remove all files beginning with "dataset"
`[xyz]` matches any character in the brackets (x, y, or z). example: `cat do[or]m.txt` will display the contents of either doom.txt or dorm.txt

The CLASSE Linux System

Getting comfortable in the terminal

- Use tab completion so you don't have to type out long file names or commands
- *DO* make liberal use of your favorite search engine and / or AI chatbot...***but DO NOT copy & paste commands without understanding what they do and how they work.***
- `nano` is a good option for editing files in the terminal (`emacs` and `vi` are also available and can be more powerful, but have a steeper learning curve)
- `atom` and `gedit` are good options for editing files with a GUI on CLASSE Linux machines

The CLASSE Linux System

Summary

- Log in with [ssh](#) for terminal access, [NoMachine](#) for full graphical desktop
- Use `lnx201` for everyday tasks, the [CLASSE Compute Farm](#) for resource-intensive jobs
 - For hardware control & other specialty tasks, ask your project mentor
- Do not put files in your home directory!
 - CHESS students — work in `/nfs/chess/user/<CLASSE-ID>/`
- See <https://xcitecourse.org/theme2/SF100/getting-started.html> for more guided materials on logging in using nomachine and ssh
- See <https://xcitecourse.org/theme2/SF100/> for more guided materials on how to use the Linux terminal

The CLASSE Linux System

Exercise 4

1. Log on to Inx201

```
ssh kls286@inx201.classe.cornell.edu
```

2. Change to a working directory

```
cd /nfs/chess/user/kls286/demo
```

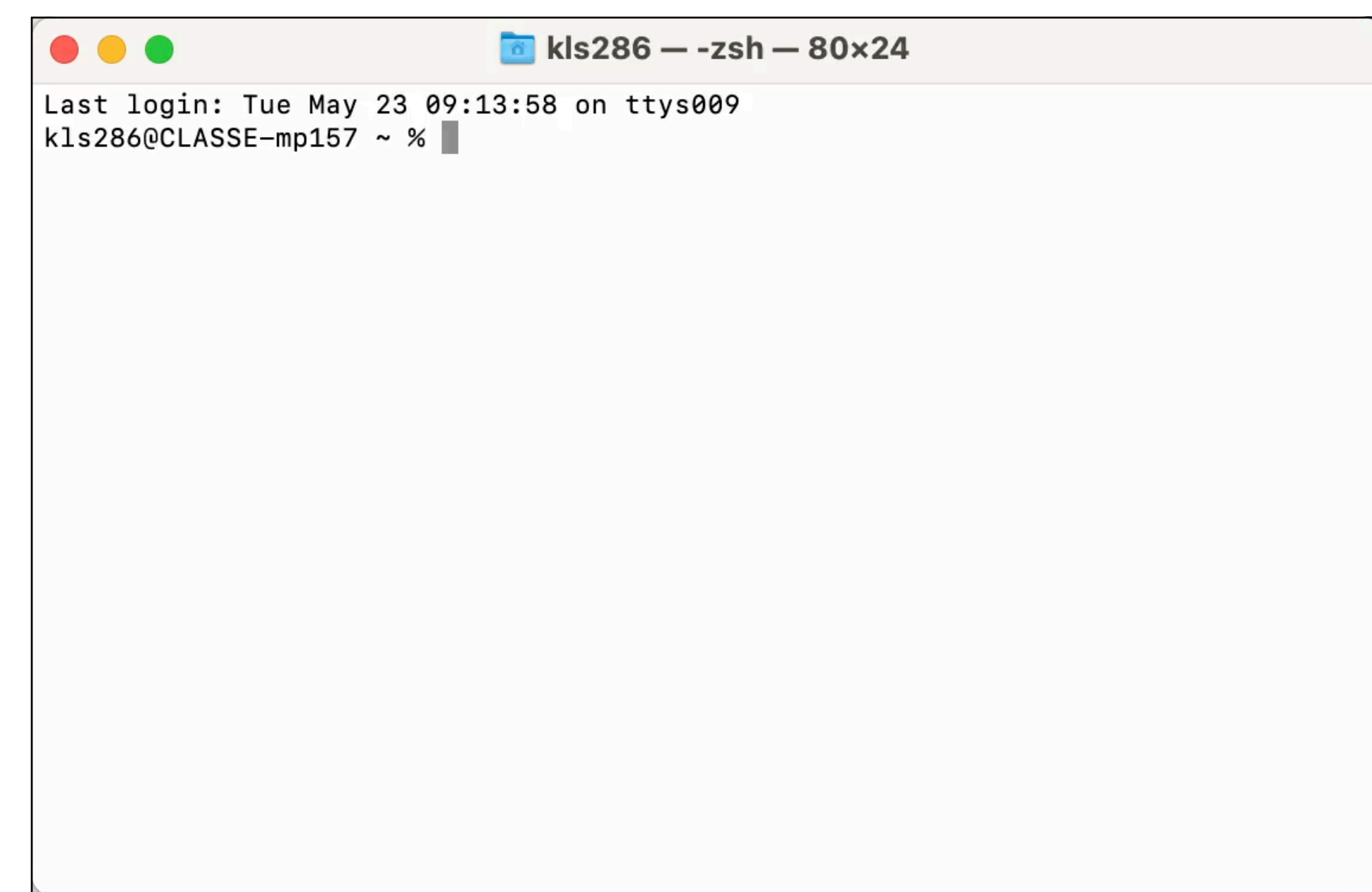
3. Use `nano` to write a “helloworld” script

4. Change the file mode of the script to be executable by the user who owns it

```
chmod u+x helloworld.sh
```

5. Hop to an a compute farm node for interactive jobs

```
qssh -q interactive.q
```



A terminal window titled "kls286 -- -zsh -- 80x24". The window shows the following text: "Last login: Tue May 23 09:13:58 on ttys009" and "kls286@CLASSE-mp157 ~ %". The terminal is currently empty with a cursor at the end of the prompt.

6. Change directories

```
cd /nfs/chess/user/kls286/demo
```

7. Run the script

```
./helloworld.sh
```

8. Log out of the compute farm

```
exit
```

9. Log out of Inx201

```
exit
```