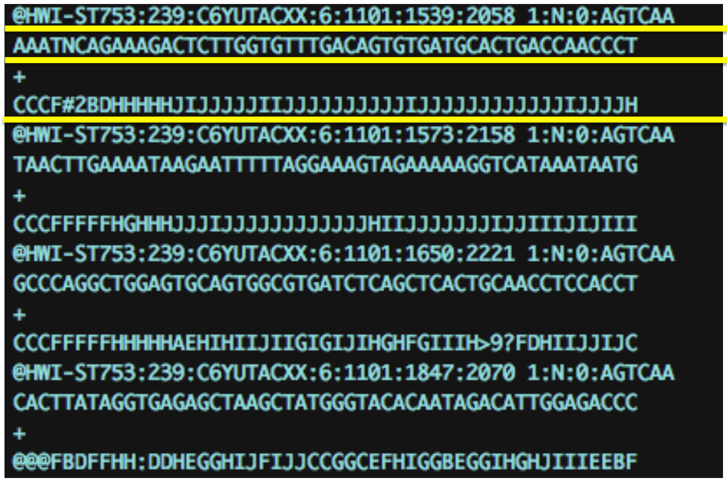
Jan 31: Job control.

What is a Fastq file?



```
@HWI-ST753:239:C6YUTACXX:6:1101:1539:2058 1:N:0:AGTCAA
AAATNCAGAAAGACTCTTGGTGTTTGACAGTGTGATGCACTGACCAACCCT
+
CCCF#2BDHHHHHJIJJJJJIIJJJJJJJJJJIJJJJJJJJJJJIJJJJH
@HWI-ST753:239:C6YUTACXX:6:1101:1573:2158 1:N:0:AGTCAA
TAACTTGAAAATAAGAATTTTTAGGAAAGTAGAAAAAGGTCATAAATAATG
+
CCCFFFFFHGHHHJJJIJJJJJJJJJJJHIIJJJJJJJIJJIIIJIJIII
@HWI-ST753:239:C6YUTACXX:6:1101:1650:2221 1:N:0:AGTCAA
GCCCAGGCTGGAGTGCAGTGGCGTGATCTCAGCTCACTGCAACCTCCACCT
+
CCCFFFFFHHHHHAEHIHIIJIIGIGIJIHGHFGIIIH>9?FDHIIJJIJC
@HWI-ST753:239:C6YUTACXX:6:1101:1847:2070 1:N:0:AGTCAA
CACTTATAGGTGAGAGCTAAGCTATGGGTACACAATAGACATTGGAGACCC
+
@@@FBDFFHH:DDHEGGHIJFIJJCCGGCEFHIGGBEGGIHGHJIIIEEBF
```

Read Identifier

Sequence

Quality

Evaluating Sequencing and Library Quality

FastQC High Throughput Sequence QC Report
Version: 0.11.2
www.bioinformatics.babraham.ac.uk/projects/
© Simon Andrews, Pierre Lindenbaum, Brian Howard, Phil Ewels 2011-14,
Picard BAM/SAM reader ©The Broad Institute, 2013
BZip decompression ©Matthew J. Francis, 2011
Base64 encoding ©Robert Harder, 2012

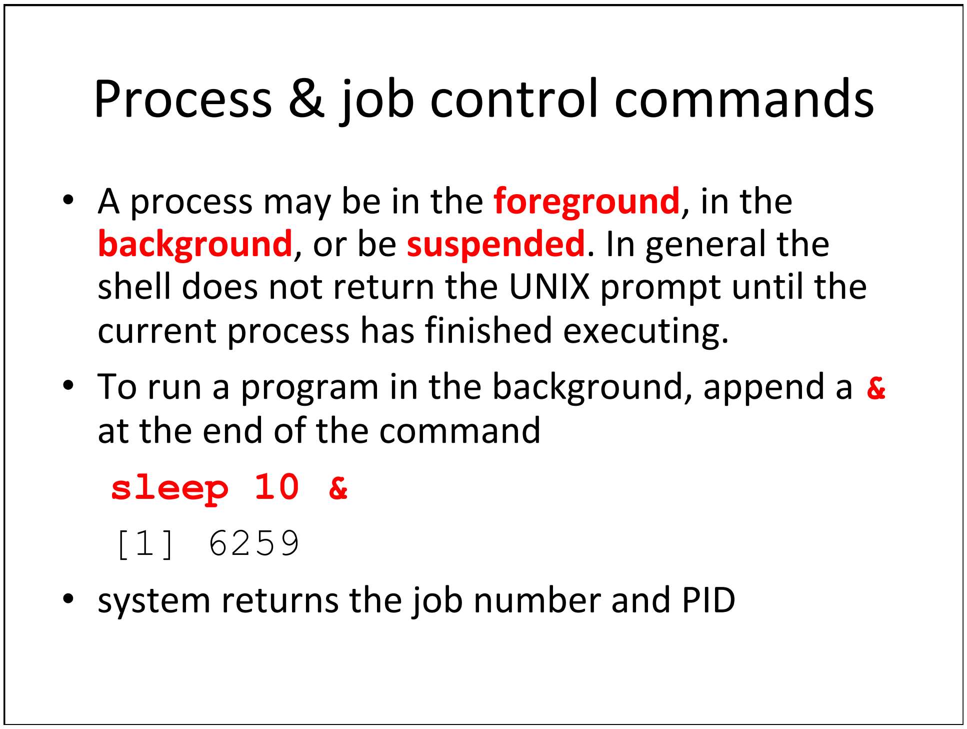
Process & job control: (head node or a single computer)

- A process is an executing program with a unique ID (PID).
- To display information about your processes with their PID and status:

```
ps
```

- to display a list of all processes on the system with full listing

```
ps –Af
```

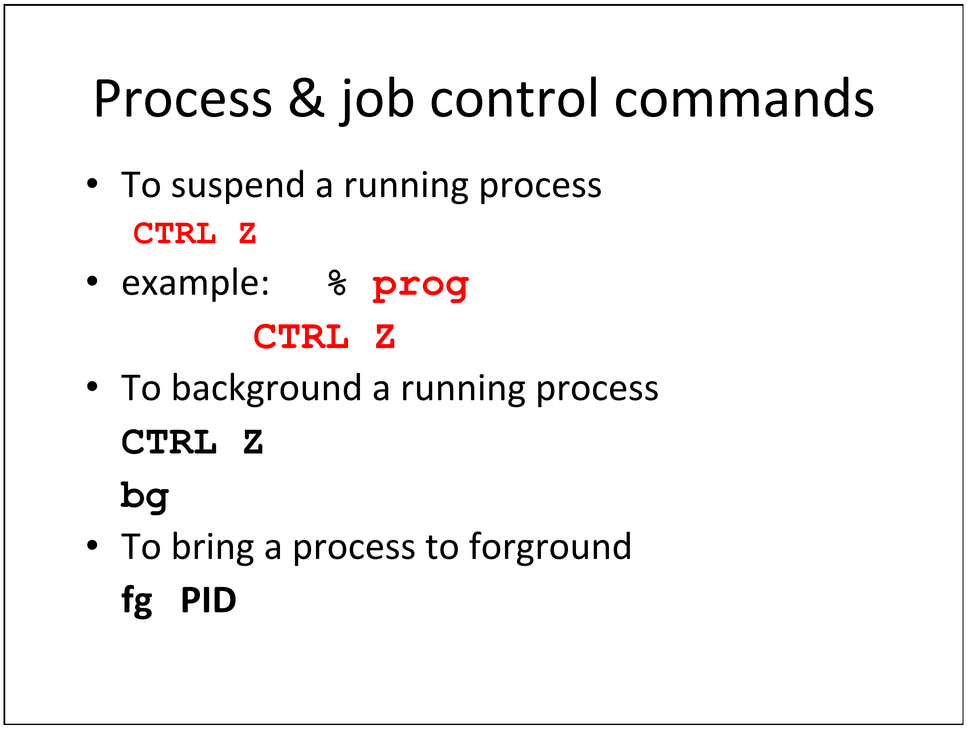


Process & job control commands

- A process may be in the **foreground**, in the **background**, or be **suspended**. In general the shell does not return the UNIX prompt until the current process has finished executing.
- To run a program in the background, append a **&** at the end of the command

```
sleep 10 &
[1] 6259
```

- system returns the job number and PID



Process & job control commands

- To suspend a running process

```
CTRL Z
```

- example: `% prog`

```
CTRL Z
```

- To background a running process

```
CTRL Z
bg
```

- To bring a process to forground

fg PID

Process & job control commands

- to kill a background process

```
kill PID
```

- to suspend a running background process

```
stop PID
```

Process & job control commands

- **Background process can not use the standard I/O. ==> Need to redirect I/O**

```
e.g:    grep mysort *.c &
```

output will be lost, instead:

```
grep mysort *.c > file1 &
```

Why do we need a BIG computer?

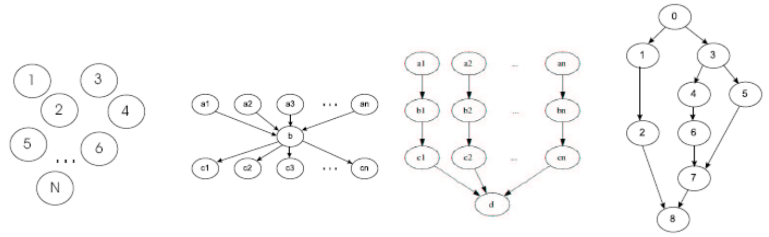
- *Compute Intensive*: A single problem requiring a large amount of computation.
- *Memory Intensive*: A single problem requiring a large amount of memory.
- *Data Intensive*: A single problem operating on a large amount of data.
- *High Throughput*: Many unrelated problems to be executed in bulk.

Basic of compute intensive problems:

- Distribute the work for a single problem across multiple CPUs to reduce the execution time as far as possible.
- Program workload must be parallelised:
  - Parallel programs split workload onto processes/threads.
  - Each process/thread performs a part of the work on its own CPU, concurrently with the others.
  - The CPUs typically need to exchange information rapidly, requiring specialized communication hardware.
  - The traditional domain of HPC and the Supercomputer
- Many tasks are **"trivially parallelizable"** meaning YOU can parallelize without special software. Little or no inter-CPU communication necessary.

Alternative classification

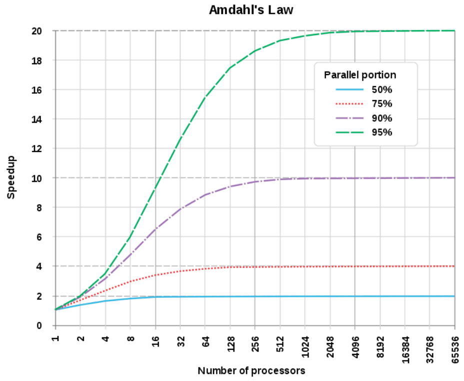
- independent tasks
- loosely-coupled tasks
- tightly-coupled tasks



(a) (b) (c) (d)

Amdahl's law (scaling)

- Using more CPUs is not necessarily faster.
  - Typically parallel codes have a scaling limit.
  - Partly due to the system overhead of managing more copies, but also to more basic constraints;



the theoretical speedup is always limited by the part of the task that cannot benefit from the improvement

Schedulers

Photo courtesy: https://www.rewardsnetwork.com/blog/maitre-d-101-10-tasks-critical-successful-restaurant-hosting/

Job Submission

- Each job must be described and given to the scheduler
- Jobs are submitted from the login nodes
  - not themselves managed by the scheduler
- Jobs are non-interactive (batch)

- The scheduler must allocate sufficient resources for the job and avoid conflicts

Job control on a cluster

- Utilize **modules** to have access to different programs (any single machine uses paths).
- To submit your job:

```
> sbatch jobscript
```

- To look at current queue:

```
> squeue          OR        > qstat
```

- To delete a job:

```
> scancel job_id
```

Using modules

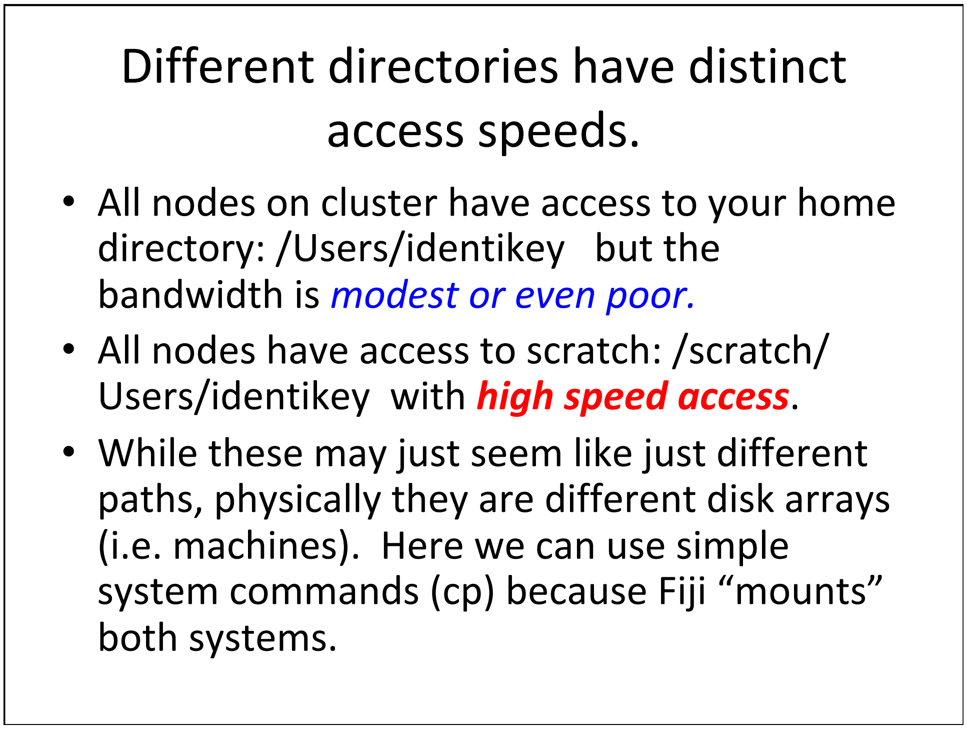
```
$ module load fastqc/0.11.5

$ fastqc --help
```

```
$ module list
```

What modules have I already loaded?

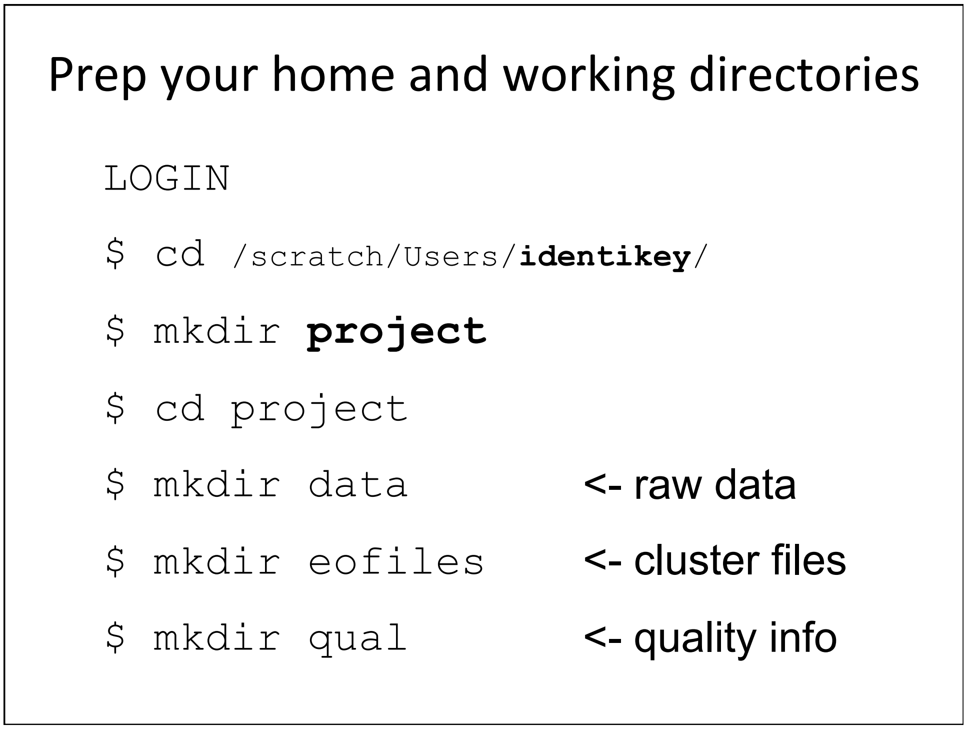
```
$ module avail
```

What modules are available for loading?



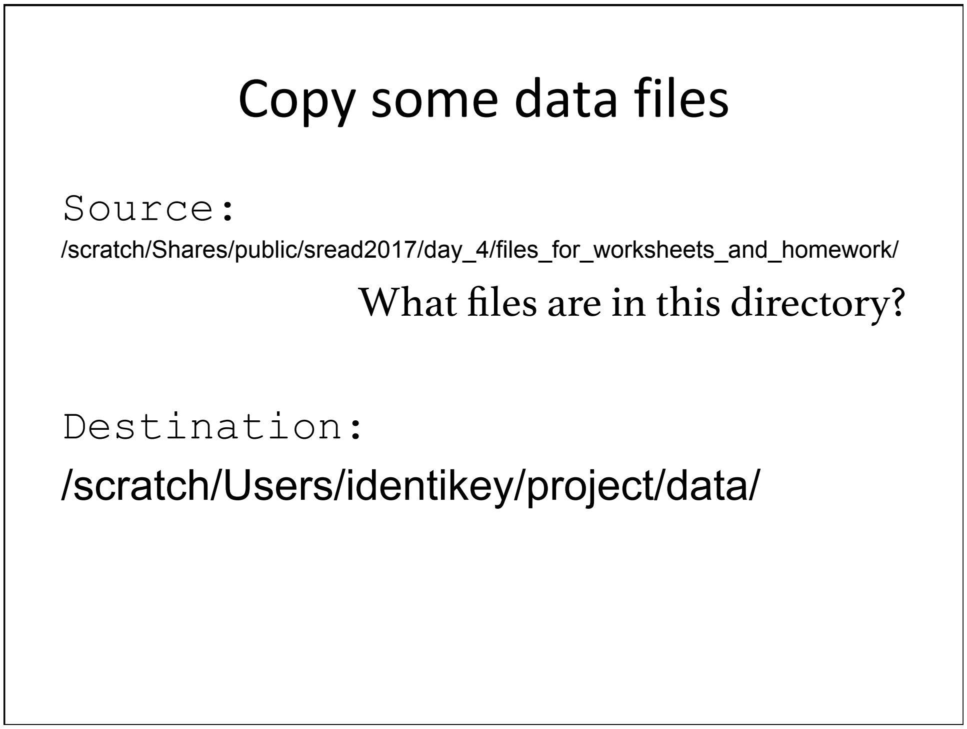
Different directories have distinct access speeds.

- All nodes on cluster have access to your home directory: /Users/identikey but the bandwidth is *modest or even poor.*
- All nodes have access to scratch: /scratch/ Users/identikey with ***high speed access***.
- While these may just seem like just different paths, physically they are different disk arrays (i.e. machines). Here we can use simple system commands (cp) because Fiji “mounts” both systems.



Prep your home and working directories

```
LOGIN

$ cd /scratch/Users/identikey/

$ mkdir project

$ cd project

$ mkdir data          <- raw data

$ mkdir eofiles       <- cluster files

$ mkdir qual          <- quality info
```



Copy some data files

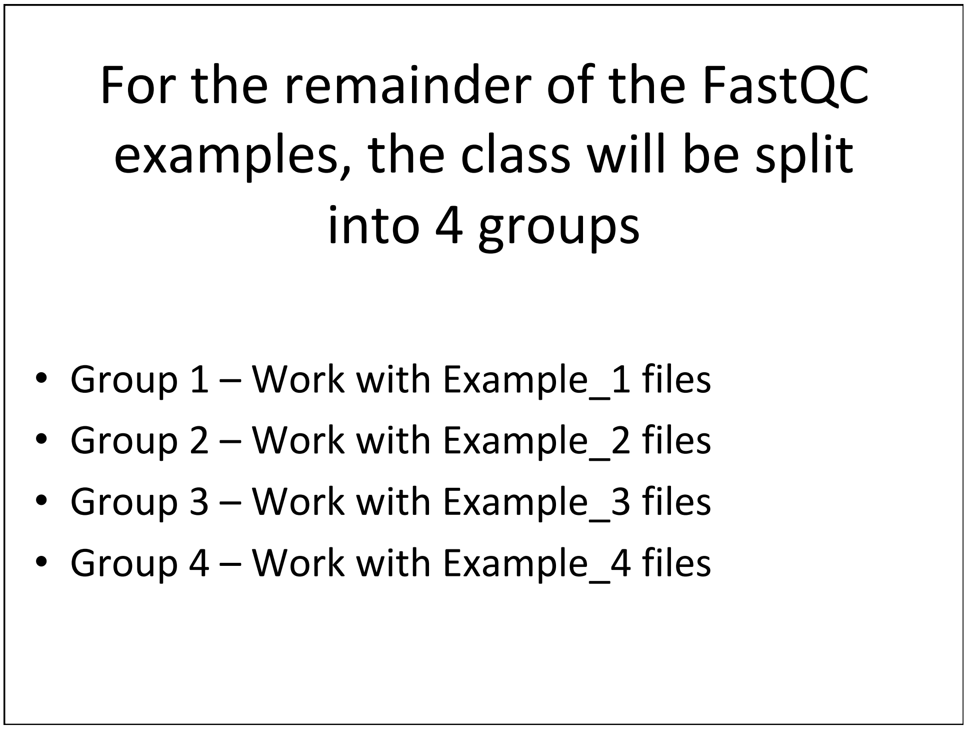
Source:

/scratch/Shares/public/sread2017/day_4/files_for_worksheets_and_homework/

What files are in this directory?

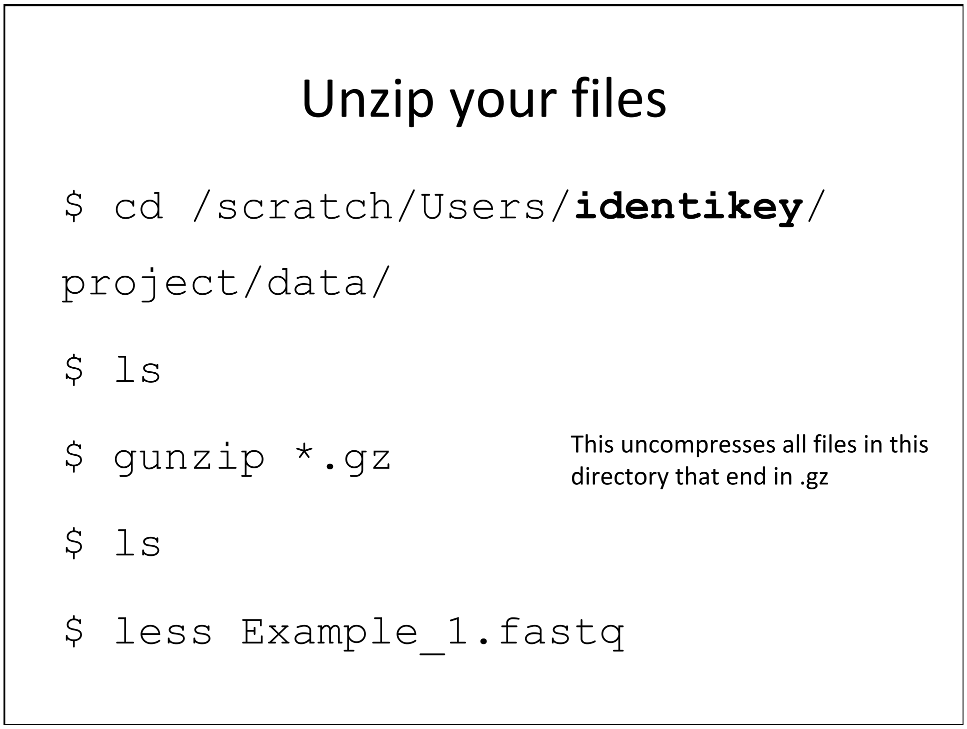
Destination:

/scratch/Users/identikey/project/data/



For the remainder of the FastQC examples, the class will be split into 4 groups

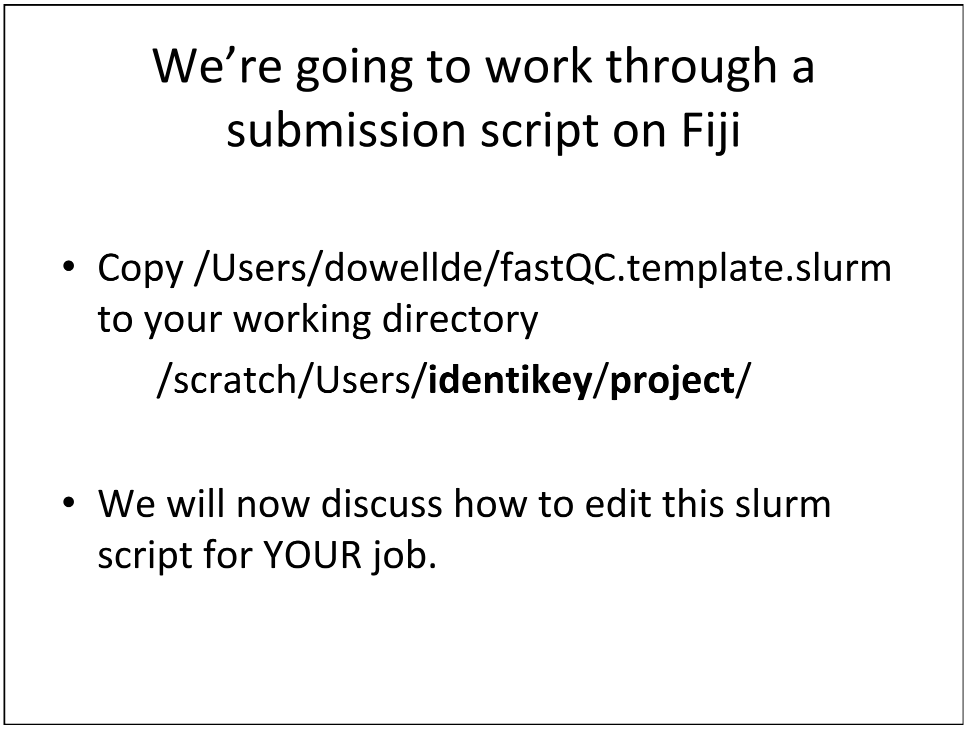
- Group 1 – Work with Example_1 files
- Group 2 – Work with Example_2 files
- Group 3 – Work with Example_3 files
- Group 4 – Work with Example_4 files



Unzip your files

```
$ cd /scratch/Users/identikey/
project/data/
$ ls
$ gunzip *.gz
$ ls
$ less Example_1.fastq
```

This uncompresses all files in this directory that end in .gz



We're going to work through a submission script on Fiji

- Copy /Users/dowellde/fastQC.template.slurm to your working directory
  /scratch/Users/**identikey**/**project**/
- We will now discuss how to edit this slurm script for YOUR job.

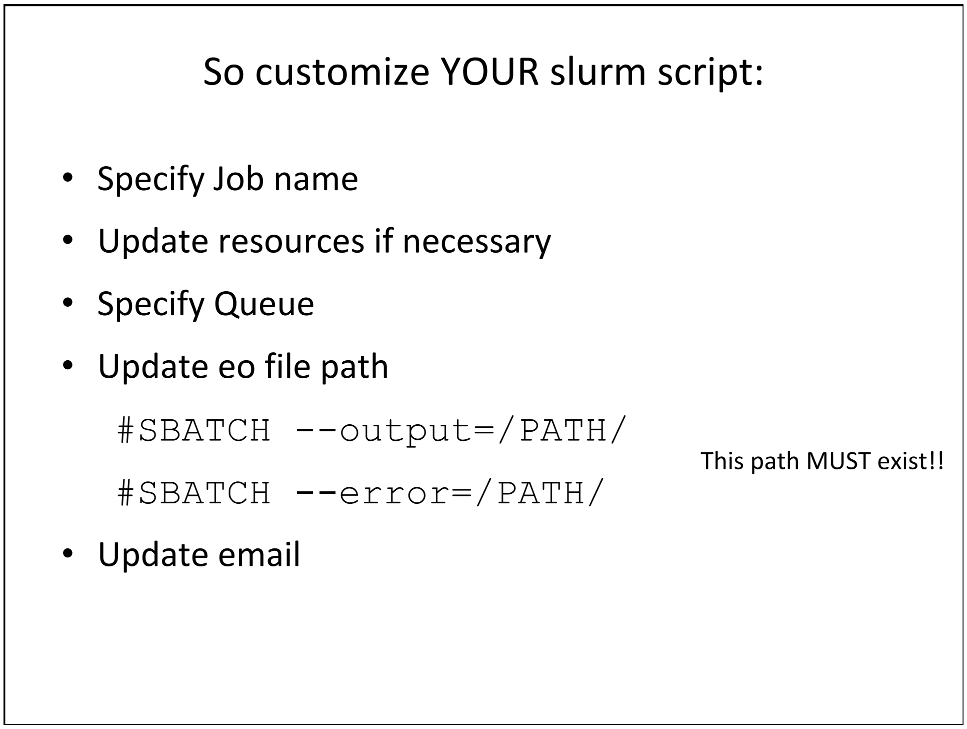
SLURM scripts

- You communicate with the scheduler via lines in the script that begin #SBATCH

- These commands generally take the form:

#SBATCH -opt

```
#!/bin/bash
#SBATCH --job-name=RDD_FASTQC # Job name
#SBATCH --mail-type=NONE # Mail events (NONE, BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=dowellde@colorado.edu# Where to send mail
#SBATCH --nodes=1  # Number of distinct nodes requested
#SBATCH --ntasks=32# Number of CPU (processer cores i.e. tasks)
#SBATCH --time=48:00:00 # Time limit hrs:min:sec
#SBATCH -p short
#SBATCH --mem=200gb # Memory limit
#SBATCH --output=/scratch/Users/dowellde/project/e_and_o/RDD_FASTQC.%j.out
#SBATCH --error=/scratch/Users/dowellde/project/e_and_o/RDD_FASTQC.%j.err
```



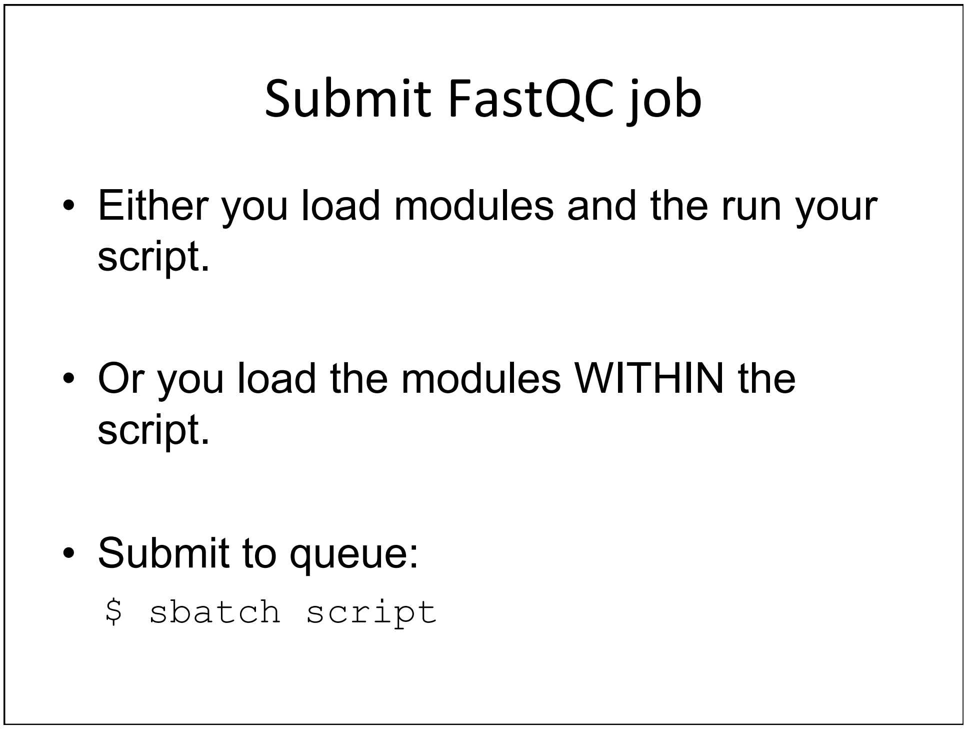
So customize YOUR slurm script:

- Specify Job name
- Update resources if necessary
- Specify Queue
- Update eo file path

```
#SBATCH --output=/PATH/
#SBATCH --error=/PATH/
```

This path MUST exist!!

- Update email



Submit FastQC job

- Either you load modules and the run your script.
- Or you load the modules WITHIN the script.
- Submit to queue:

```
$ sbatch script
```

Look at eofiles

```
$ cd eofiles
$ ls
$ less jobname.12345.err
$ less jobname.12345.out
```

Locate output files

```
$ cd /scratch/Users/identikey/project/qual
$ ls
```

Notice that FastQC outputs an HTML file and a Zipped file.

Transferring files between machines

- More generally, transferring files between one physical machine (Fiji) and another (say a public server) requires a transfer program. Most public servers run FTP (file transfer protocol).

YOU WANT YOUR COUSIN TO SEND YOU A FILE? EASY. HE CAN EMAIL IT TO— ... OH, IT'S 25 MB? HMM...
DO EITHER OF YOU HAVE AN FTP SERVER? NO, RIGHT.
IF YOU HAD WEB HOSTING, YOU COULD UPLOAD IT...
HMM. WE COULD TRY ONE OF THOSE MEGASHAREUPLOAD SITES, BUT THEY'RE FLAKY AND FULL OF DELAYS AND PORN POPUPS.
HOW ABOUT AIM DIRECT CONNECT? ANYONE STILL USE THAT?
OH, WAIT, DROPBOX! IT'S THIS RECENT STARTUP FROM A FEW YEARS BACK THAT SYNCS FOLDERS BETWEEN COMPUTERS. YOU JUST NEED TO MAKE AN ACCOUNT, INSTALL THE—
OH, HE JUST DROVE OVER TO YOUR HOUSE WITH A USB DRIVE?
UH, COOL, THAT WORKS, TOO.
I LIKE HOW WE'VE HAD THE INTERNET FOR DECADES, YET "SENDING FILES" IS SOMETHING EARLY ADOPTERS ARE STILL FIGURING OUT HOW TO DO.

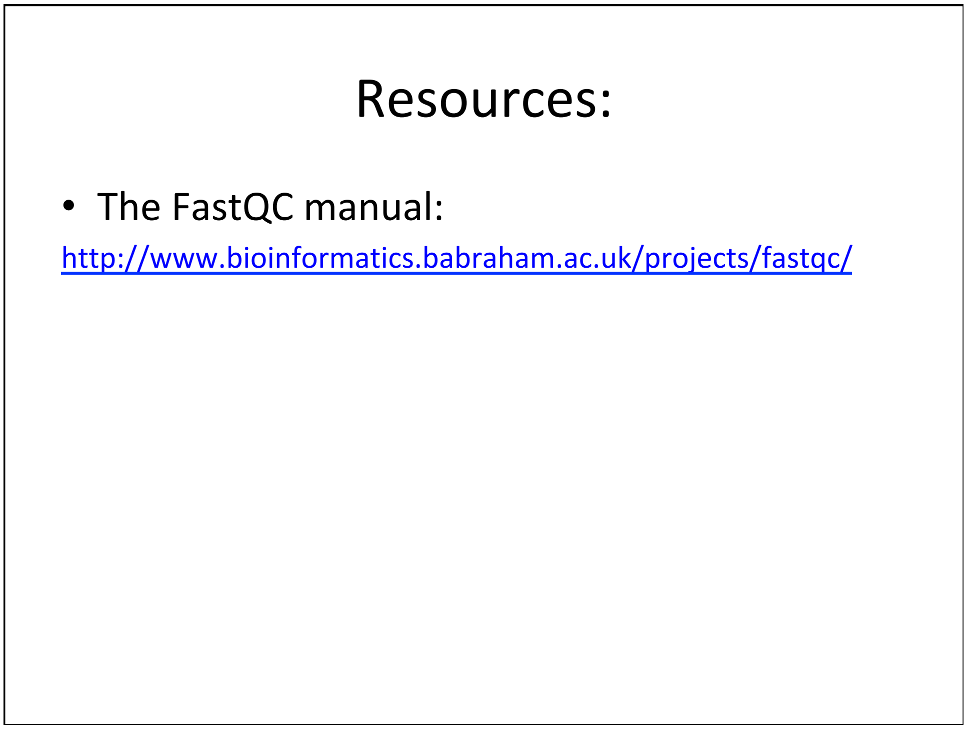
…We will use secure File Transfer Protocols (SFTP and SCP)

Transferring files between machines

- More generally, transferring files between one physical machine (Fiji) and another (say your laptop) requires a ***secure*** copy program. The Linux command is scp. We will be running this program ON YOUR LAPTOP.
- THEREFORE: Windows users will need to identify and install a free scp software package (e.g. WinSCP or equivalent)

Assignment this week:

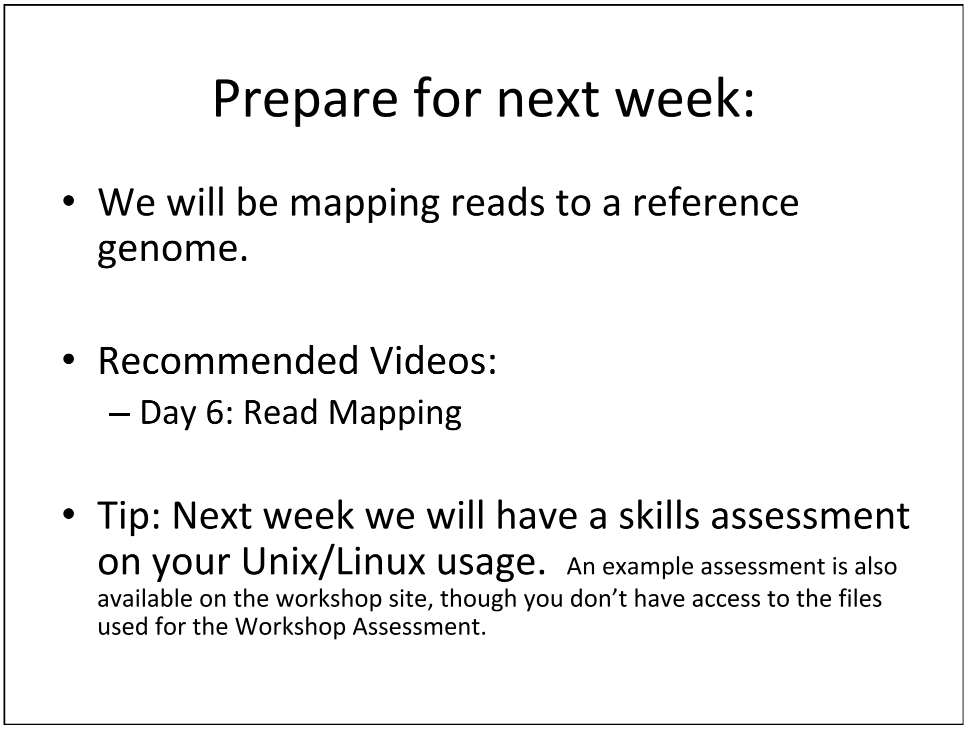
- You will revise your Fiji script to run FastQC on a different sequencing dataset.
- You will transfer the output to your local laptop and examine the results.
- Be prepared next week to share with the class what you learned about the library analyzed.



Resources:

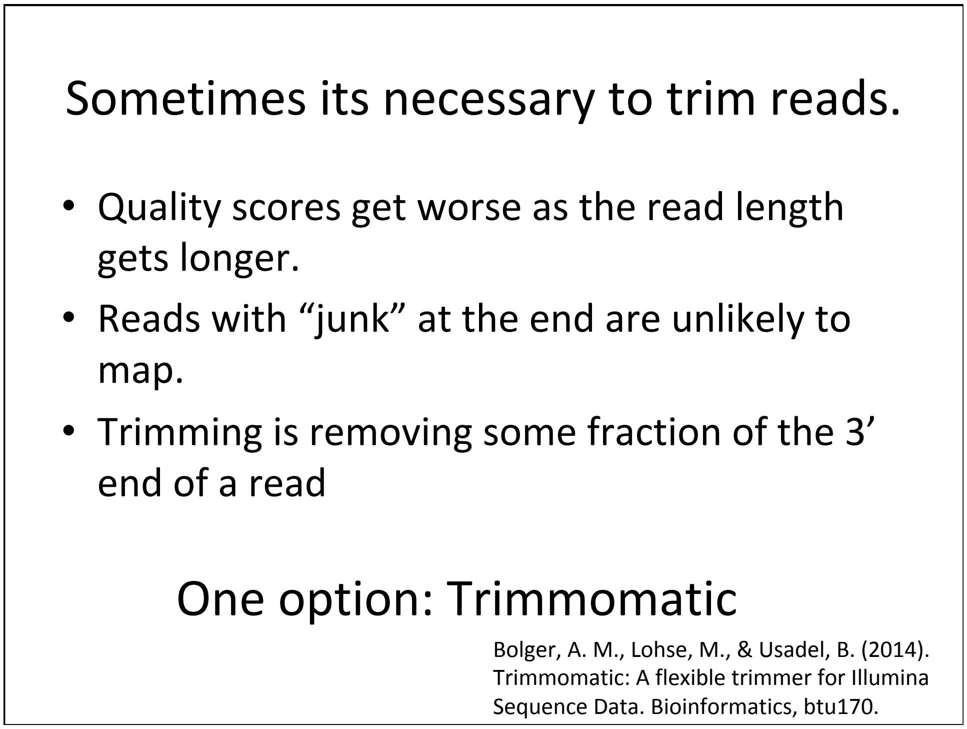
- The FastQC manual:

http://www.bioinformatics.babraham.ac.uk/projects/fastqc/



Prepare for next week:

- We will be mapping reads to a reference genome.
- Recommended Videos:
  - Day 6: Read Mapping
- Tip: Next week we will have a skills assessment on your Unix/Linux usage. An example assessment is also available on the workshop site, though you don’t have access to the files used for the Workshop Assessment.

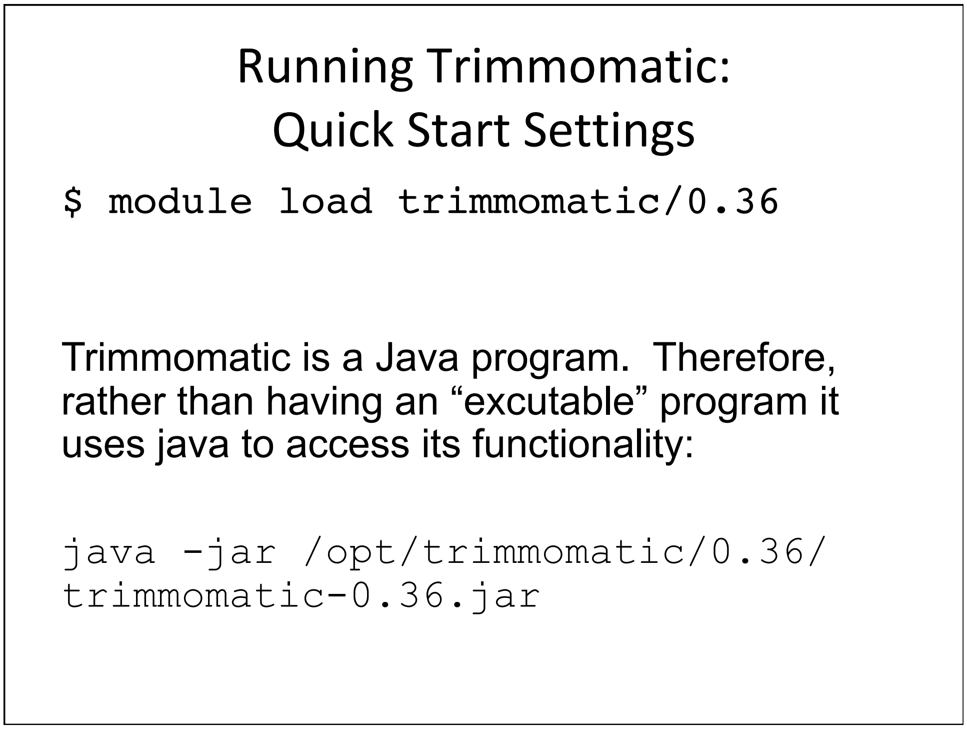


Sometimes its necessary to trim reads.

- Quality scores get worse as the read length gets longer.
- Reads with “junk” at the end are unlikely to map.
- Trimming is removing some fraction of the 3’ end of a read

One option: Trimmomatic

Bolger, A. M., Lohse, M., & Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina Sequence Data. Bioinformatics, btu170.

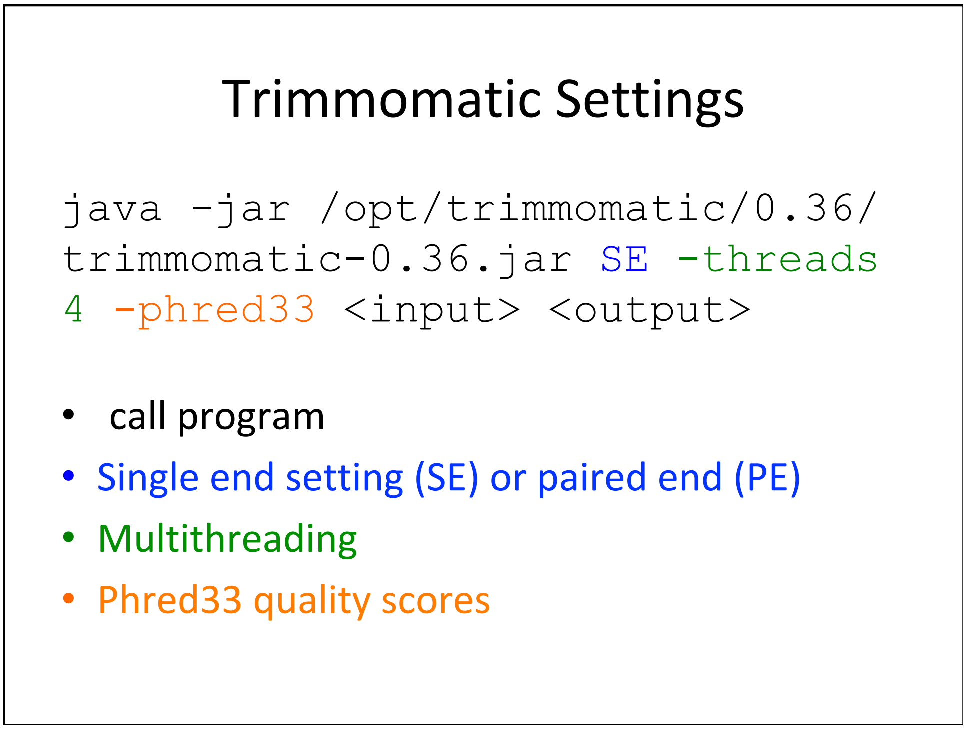


Running Trimmomatic: Quick Start Settings

```
$ module load trimmomatic/0.36
```

Trimmomatic is a Java program. Therefore, rather than having an “excutable” program it uses java to access its functionality:

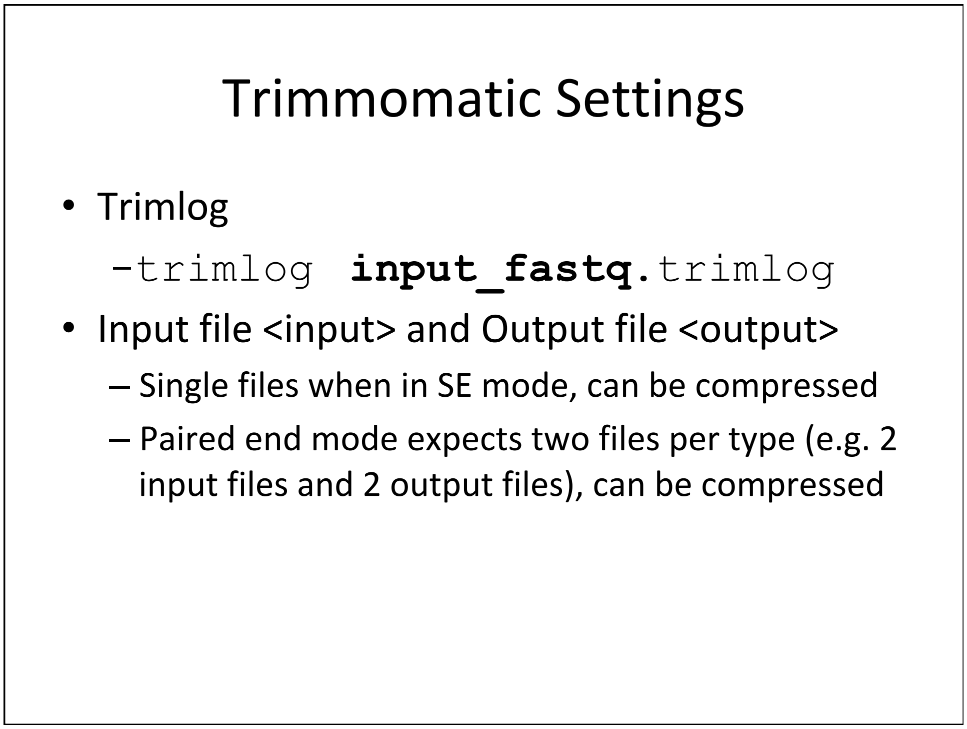
```
java -jar /opt/trimmomatic/0.36/
trimmomatic-0.36.jar
```



Trimmomatic Settings

```
java -jar /opt/trimmomatic/0.36/
trimmomatic-0.36.jar SE -threads
4 -phred33 <input> <output>
```

- call program
- Single end setting (SE) or paired end (PE)
- Multithreading
- Phred33 quality scores

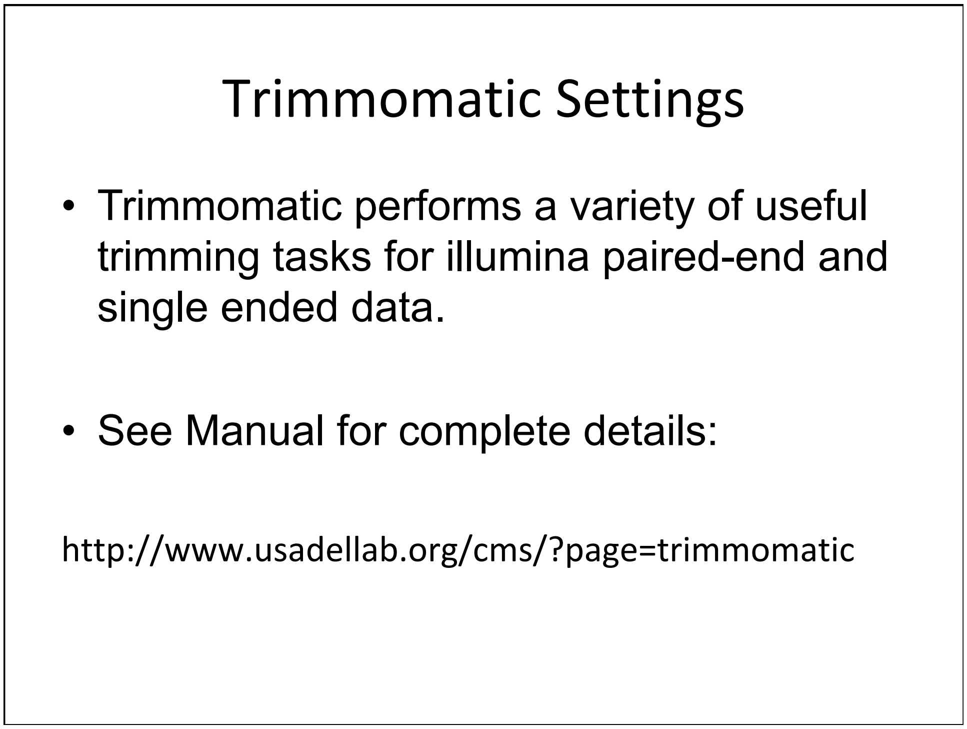


Trimmomatic Settings

- Trimlog

  ```
  -trimlog  input_fastq.trimlog
  ```

- Input file <input> and Output file <output>
  - Single files when in SE mode, can be compressed
  - Paired end mode expects two files per type (e.g. 2 input files and 2 output files), can be compressed



Trimmomatic Settings

- Trimmomatic performs a variety of useful trimming tasks for illumina paired-end and single ended data.

- See Manual for complete details:

http://www.usadellab.org/cms/?page=trimmomatic