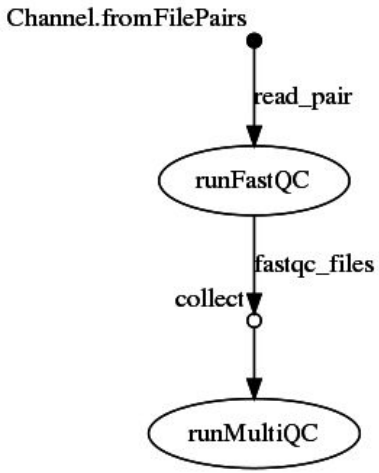
Nextflow intro

Gerrit Botha
Ilifu advanced training
University of Cape Town, Cape Town, South Africa
May 2020

Background

- DSL (Groovy extension) and workflow engine.
- Fast prototyping - simpler to put together tasks and use existing scripts
- Supported for various batch schedulers (SGE, LSF, PBS, SLURM). Also support for AWS batch, GCP and Kubernetes.
- Ideal for high throughput problems. Split by sample, chromosome, position, iterate over a range of variables.
- Singularity and docker support.
- Job walltime, memory and core requirements can be specified.
- Good documentation and user support.
- We will use the example here to go through Nextflow features.



Processes and channels

- A nextflow script is made by joining different processes. Processes are connected to each other with channels.

```
#!/usr/bin/env nextflow

raw_reads = params.rawReads
out_dir = file(params.outDir)

out_dir.mkdir()

read_pair = Channel.fromFilePairs("${raw_reads}/*R[1,2].fastq.gz", type: 'file')

process runFastQC{
    tag { "${params.projectName}.rFQC.${sample}" }
    cpus { 2 }
    publishDir "${out_dir}/qc/raw/${sample}", mode: 'copy', overwrite: false

    input:
        set sample, file(in_fastq) from read_pair

    output:
        file("${sample}_fastqc/*.zip") into fastqc_files

    """
    mkdir ${sample}_fastqc
    fastqc --outdir ${sample}_fastqc \
    -t ${task.cpus} \
    ${in_fastq.get(0)} \
    ${in_fastq.get(1)}
    """
}
```

Operators

- Operators allows you to connect channels together or transform channels

```
process runMultiQC{
    tag { "${params.projectName}.rMQC" }
    publishDir "${out_dir}/qc/raw", mode: 'copy', overwrite: false

    input:
        file('*') from fastqc_files.collect()

    output:
        file('multiqc_report.html')

    """
    multiqc .
    """
}
```

Scripting

- The Nextflow syntax has been developed to ease the writing of computational pipelines.
- Nextflow can execute any piece of Groovy code or use any library for the JVM platform.

Variables

```
x = 1
println x

x = new
java.util.Date()
println x

x = -3.1499392
println x

x = false
println x

x = "Hi"
println x
```

Lists

```
myList = [1776, -1, 33, 99, 0,
928734928763]
println myList[0]
println myList.size()
```

Maps

```
scores = [ "Brett":100, "Pete":"Did
not finish", "Andrew":86.87934 ]
println scores["Pete"]
println scores.Pete
scores["Pete"] = 3
scores["Cedric"] = 120
```

Closure

```
square = { it * it }
println square(9)
81
[ 1, 2, 3, 4
].collect(square)
[ 1, 4, 9, 16 ]
```

- Conditional execution
- Regular expression
- File I/O

Configuration

- Pipeline configuration properties are defined in the `nextflow.config` file

```
executor{
    jobName = { "$task.tag" }
}

params {

    projectName = "tl"

    rawReads = "/ceph/cbio/datasets/16s/dog-stool/test"
    outDir = "/ceph/cbio/users/gerrit/scratch/16s/nextflow-out"

    queue = 'Main'

}

profiles{
    standard {
        process.executor = 'local'
    }

   slurm {
        process.executor = 'slurm'
        process.queue = params.queue
        process.container = 'docker://quay.io/h3abionet_org/h3a16s-fastqc'
        singularity.enabled = true
        process.memory =  4.GB
        process.cpus = 1
    }
}
```

Reporting

- Execution report (`-with-report`) - HTML report with total number of jobs and their individual and summary reports for memory, CPU, I/O usage and execution times.
- Trace report (`-with-trace`) - TSV file with individual job reports on memory, CPU, I/O usage and execution time.
- Timeline report (`-with-timeline`) - HTML timeline for all jobs executed in the pipeline.
- DAG visualisation (`-with-dag`) - DAG representation of the workflow.
- Nextflow Tower - Real time visualisation of workflow progress and resource usage.

Caching and resuming

- The caching feature generates a unique key by indexing the process script and inputs. This key is used identify unique outputs produced by the process execution.

- With the `-resume` option a workflow execution can be continued from where it got interrupted.

Nextflow setup on Ilifu

- Nextflow and Java path

```
# Path exports
export
PATH=$PATH:/ceph/cbio/soft/jdk-11.0.2/bin:/ceph/cbio/soft/nextflow/

# Environmental variable exports
export JAVA_CMD=/ceph/cbio/soft/jdk-11.0.2/bin/java
export JAVA_HOME=/ceph/cbio/soft/jdk-11.0.2
```

- Run as interactive node, test memory

```
srun --nodes=1 --ntasks=1 --mem=48g --pty bash
```

- Use /scratch as far as possible for the working directory

- Limit workflow to 100 jobs in the queue and a submission rate of 10 jobs per minute

```
 queueSize = 100
submitRateLimit = '10 min' // submit 10 jobs per minute
```

Example - Run

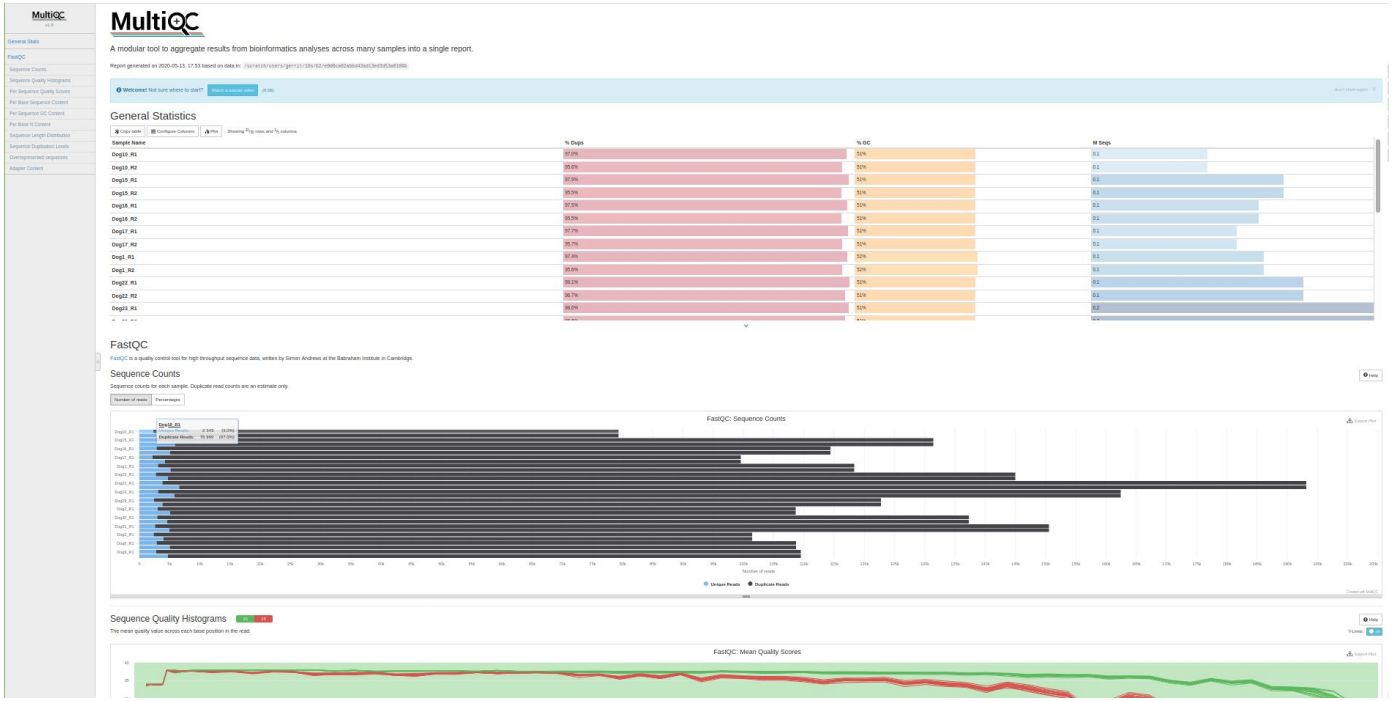
```
(base) gerrit@slurm-login:~/code/run-fastqc$ srun --nodes=1 --ntasks=1 --mem=4gb -t 2:00:00 --pty bash
(base) gerrit@slwrk-109:~/code/run-fastqc$ nextflow -log nextflow.log run -c nextflow.config  -w /scratch/users/gerrit/16s/nextflow-work main.nf -profile slurm -with-trace trace.txt -with-report report.html -with-timeline timeline.html -w
ith-dag dag.dot
N E X T F L O W  ~  version 19.10.0
Launching `main.nf` [distraught_hopper] - revision: bd0e7b15fc
executor >  slurm (16)
[a3/f376d5] process > runFastQC (tl.rFQC.Dog3) [100%] 15 of 15 ✔
[12/59c8ea] process > runMultiQC (tl.rMQC)      [  0%] 0 of 1
Pulling Singularity image docker://quay.io/h3abionet_org/h3a16s-fastqc [cache /scratch/users/gerrit/16s/nextflow-work/singularity/quay.io-h3abionet_org-h3a16s-fastqc.img]
WARN: Singularity cache directory has not been defined -- Remote image will be stored in the path: /scratch/users/gerrit/16s/nextflow-work/singularity

       Pipeline execution summary
       ---------------------------
       Completed at: 2020-05-14T20:11:16.561+02:00
executor >  slurm (16)
[a3/f376d5] process > runFastQC (tl.rFQC.Dog3) [100%] 15 of 15 ✔
[12/59c8ea] process > runMultiQC (tl.rMQC)     [100%] 1 of 1 ✔

(base) gerrit@slwrk-109:~/code/run-fastqc$ nextflow -log nextflow.log run -c nextflow.config  -w /scratch/users/gerrit/16s/nextflow-work main.nf -profile slurm -with-trace trace.txt -with-report report.html -with-timeline timeline.html -w
ith-dag dag.dot -resume
N E X T F L O W  ~  version 19.10.0
Launching `main.nf` [shrivelled_sinoussi] - revision: bd0e7b15fc
[68/6d4206] process > runFastQC (tl.rFQC.Dog24) [100%] 15 of 15, cached: 15 ✔
[12/59c8ea] process > runMultiQC (tl.rMQC)      [100%] 1 of 1, cached: 1 ✔

       Pipeline execution summary
       ---------------------------
       Completed at: 2020-05-14T20:11:44.999+02:00
       Duration    : 6.9s
       Success     : true
       workDir     : /scratch/users/gerrit/16s/nextflow-work
[68/6d4206] process > runFastQC (tl.rFQC.Dog24) [100%] 15 of 15, cached: 15 ✔
[12/59c8ea] process > runMultiQC (tl.rMQC)      [100%] 1 of 1, cached: 1 ✔
```

```
(base) gerrit@slurm-login:~/projects/16S-run-times/run/2-slurm$ squeue | grep rFQC
1289135      Main    tl.rFQC.Dog17   gerrit  R       0:06      1 2 slwrk-103
1289136      Main    tl.rFQC.Dog15   gerrit  R       0:06      1 2 slwrk-131
1289137      Main     tl.rFQC.Dog1   gerrit  R       0:06      1 2 slwrk-130
1289138      Main    tl.rFQC.Dog30   gerrit  R       0:06      1 2 slwrk-130
1289139      Main    tl.rFQC.Dog16   gerrit  R       0:06      1 2 slwrk-130
1289140      Main     tl.rFQC.Dog9   gerrit  R       0:06      1 2 slwrk-130
1289141      Main    tl.rFQC.Dog23   gerrit  R       0:06      1 2 slwrk-130
1289142      Main     tl.rFQC.Dog8   gerrit  R       0:06      1 2 slwrk-130
1289130      Main    tl.rFQC.Dog31   gerrit  R       0:07      1 2 slwrk-103
1289131      Main    tl.rFQC.Dog29   gerrit  R       0:07      1 2 slwrk-103
1289132      Main    tl.rFQC.Dog10   gerrit  R       0:07      1 2 slwrk-103
1289133      Main     tl.rFQC.Dog2   gerrit  R       0:07      1 2 slwrk-103
1289134      Main     tl.rFQC.Dog3   gerrit  R       0:07      1 2 slwrk-103
1289128      Main    tl.rFQC.Dog24   gerrit  R       0:10      1 2 slwrk-103
1289129      Main    tl.rFQC.Dog22   gerrit  R       0:10      1 2 slwrk-103
```

Example - Output

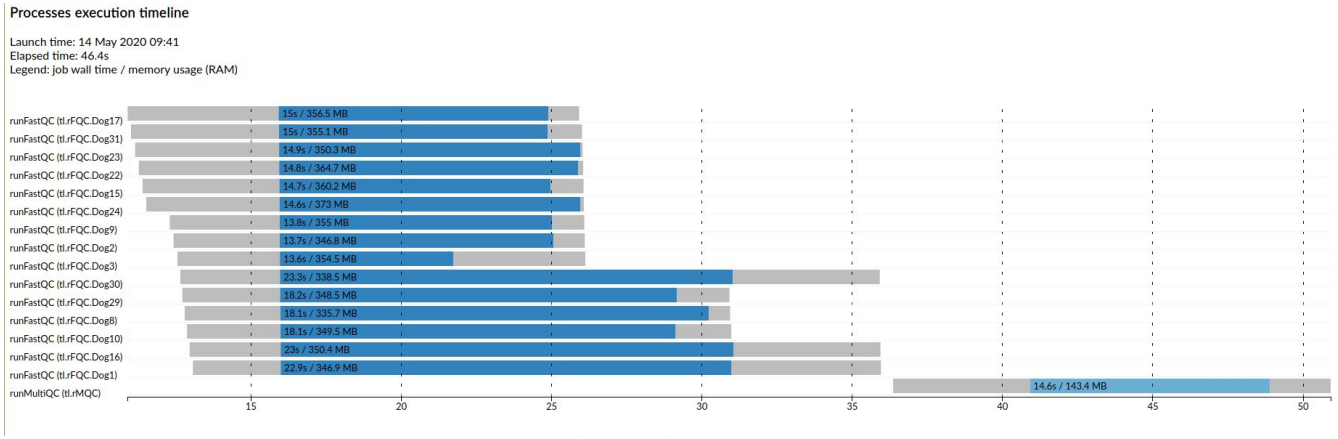
```
(base) gerrit@slwrk-133:~/code/run-fastqc$ ls -a /cbio/users/gerrit/scratch/16s/nextflow-out/qc/raw/
.  ..  Dog1  Dog10  Dog15  Dog16  Dog17  Dog2  Dog22  Dog23  Dog24  Dog29  Dog3  Dog30  Dog31  Dog8  Dog9  multiqc_report.html
(base) gerrit@slwrk-133:~/code/run-fastqc$ ls -a /cbio/users/gerrit/scratch/16s/nextflow-out/qc/raw/Dog1
Dog1/  Dog10/ Dog15/ Dog16/ Dog17/
(base) gerrit@slwrk-133:~/code/run-fastqc$ ls -a /cbio/users/gerrit/scratch/16s/nextflow-out/qc/raw/Dog1/Dog1_fastqc/
.  ..  Dog1_R1_fastqc.zip  Dog1_R2_fastqc.zip
```

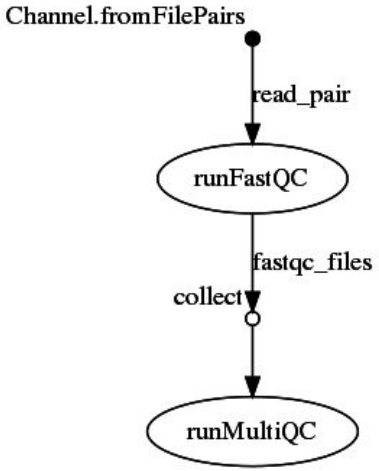
```
(base) gerrit@slwrk-133:~/code/run-fastqc$ ls -a /scratch/users/gerrit/16s/nextflow-work/
.  ..  0a  2b  40  4e  59  73  77  85  92  ba  bc  bd  cc  de  e1  singularity
(base) gerrit@slwrk-133:~/code/run-fastqc$ ls -a /scratch/users/gerrit/16s/nextflow-work/singularity/
.  ..  quay.io-h3abionet_org-h3a16s-fastqc.img
(base) gerrit@slwrk-133:~/code/run-fastqc$ ls -a /scratch/users/gerrit/16s/nextflow-work/0a/2bb515e7ecb9191d7f244cae7697b4/
.  ..  .command.begin  .command.err  .command.log  .command.out  .command.run  .command.sh  .command.trace  .exitcode  Dog10_R1.fastq.gz  Dog10_R2.fastq.gz  Dog10_fastqc
(base) gerrit@slwrk-133:~/code/run-fastqc$ cat /scratch/users/gerrit/16s/nextflow-work/0a/2bb515e7ecb9191d7f244cae7697b4/.command.sh
#!/bin/bash -ue
mkdir Dog10_fastqc
fastqc --outdir Dog10_fastqc     -t 2     Dog10_R1.fastq.gz     Dog10_R2.fastq.gz
(base) gerrit@slwrk-133:~/code/run-fastqc$
```

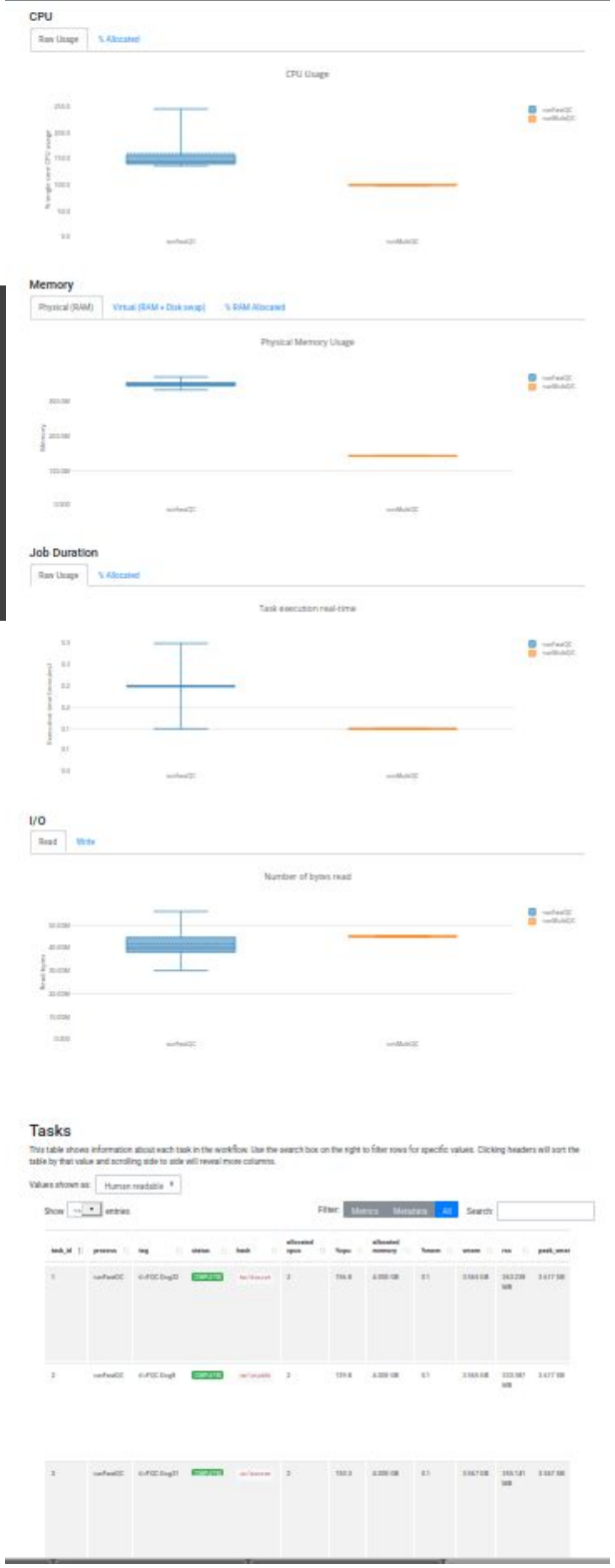


Example - Reports

```
(base) gerrit@slwrk-133:~/code/run-fastqc$ cat trace.txt | column -s$'\t' -t
task_id  hash       native_id  name                       status     exit  submit                  duration  realtime  %cpu    peak_rss  peak_vmem  rchar    wchar
13       2b/bd94e0  1289145    runFastQC (tl.rFQC.Dog17)  COMPLETED  0     2020-05-14 09:41:10.895  15s       9s        140.8%  356.5 MB  3.6 GB     35.8 MB  4 MB
3        4e/032e30  1289146    runFastQC (tl.rFQC.Dog31)  COMPLETED  0     2020-05-14 09:41:11.008  15s       8.9s      150.3%  355.1 MB  3.6 GB     47.3 MB  4 MB
15       ba/e9e06f  1289147    runFastQC (tl.rFQC.Dog23)  COMPLETED  0     2020-05-14 09:41:11.144  14.9s     10s       150.9%  350.3 MB  3.6 GB     56.1 MB  4 MB
1        59/d19440  1289148    runFastQC (tl.rFQC.Dog22)  COMPLETED  0     2020-05-14 09:41:11.268  14.8s     10s       156.8%  364.7 MB  3.6 GB     45.6 MB  4.1 MB
14       85/9695c4  1289149    runFastQC (tl.rFQC.Dog15)  COMPLETED  0     2020-05-14 09:41:11.392  14.7s     9s        144.1%  360.2 MB  3.6 GB     43.9 MB  4.1 MB
5        bc/a4cf68  1289150    runFastQC (tl.rFQC.Dog24)  COMPLETED  0     2020-05-14 09:41:11.517  14.6s     10s       163.5%  373 MB    3.6 GB     49.7 MB  4 MB
8        bd/bf4551  1289151    runFastQC (tl.rFQC.Dog9)   COMPLETED  0     2020-05-14 09:41:12.296  13.8s     9.1s      140.7%  355 MB    3.6 GB     38.1 MB  4 MB
4        e1/432a62  1289152    runFastQC (tl.rFQC.Dog2)   COMPLETED  0     2020-05-14 09:41:12.422  13.7s     9.1s      136.8%  346.8 MB  3.6 GB     38.5 MB  4.1 MB
10       73/81ccef  1289153    runFastQC (tl.rFQC.Dog3)   COMPLETED  0     2020-05-14 09:41:12.555  13.6s     5.8s      245.6%  354.5 MB  3.6 GB     35.9 MB  4 MB
12       77/364769  1289155    runFastQC (tl.rFQC.Dog29)  COMPLETED  0     2020-05-14 09:41:12.717  18.2s     13.2s     163.0%  348.5 MB  3.6 GB     40.1 MB  4.2 MB
2        40/1e4dd8  1289156    runFastQC (tl.rFQC.Dog8)   COMPLETED  0     2020-05-14 09:41:12.795  18.1s     14.3s     139.8%  335.7 MB  3.6 GB     38.6 MB  4 MB
9        0a/2bb515  1289157    runFastQC (tl.rFQC.Dog10)  COMPLETED  0     2020-05-14 09:41:12.870  18.1s     13.1s     157.0%  349.5 MB  3.6 GB     30.3 MB  4 MB
6        de/c11f27  1289154    runFastQC (tl.rFQC.Dog30)  COMPLETED  0     2020-05-14 09:41:12.649  23.3s     15.1s     139.9%  338.5 MB  3.6 GB     42.4 MB  4 MB
11       92/83841a  1289158    runFastQC (tl.rFQC.Dog16)  COMPLETED  0     2020-05-14 09:41:12.957  23s       15.1s     141.3%  350.4 MB  3.6 GB     39.3 MB  4 MB
7        92/41c154  1289159    runFastQC (tl.rFQC.Dog1)   COMPLETED  0     2020-05-14 09:41:13.068  22.9s     15s       137.8%  346.9 MB  3.6 GB     40.8 MB  4.1 MB
16       cc/d2e906  1289160    runMultiQC (tl.rMQC)       COMPLETED  0     2020-05-14 09:41:36.361  14.6s     8s        99.8%   143.4 MB  1.4 GB     45.2 MB  5.7 MB
```







Extra features

- `scratch` directive allows you to process the working directory on a tmp directory that is local to the node. When the `ramdisk` string is provided to `scratch` the local memory of the execution node will be used as scratch space.

- DSL v2 allows for modularisation

```
nextflow.preview.dsl=2

include 'modules/align'
include 'modules/mark-duplicates'
include 'modules/bqsr'

Channel
    .from( 'id.fastq' )
    .set { fastq }

workflow {
    align(fastq)
    mark_duplicates(align.output)
    bqsr(mark_duplicates.output)
}
```

```
// Need to define variable
def fastq = ""

process align {

    publishDir 'results', mode:'copy'

    input:
    file (f) from fastq

    output:
    file ("${f.baseName}.align.bam")

    script:
    """
    touch "${f.baseName}.align.bam"
    """
}
```

Documentation

- Latest documentation: https://www.nextflow.io/docs/latest/index.html
- Working examples: https://github.com/nextflow-io/awesome-nextflow
- Workflows for standard protocols: https://github.com/nf-core
- Recurrent implementation patterns used in Nextflow applications: https://nextflow-io.github.io/patterns/index.html