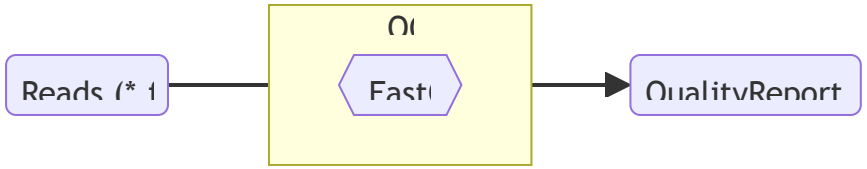
CWL in ARCs

DataPLANT Data Steward Circle – Feb 5th, 2025

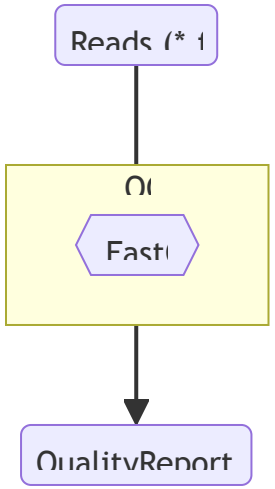
Dominik Brilhaus, CEPLAS Data

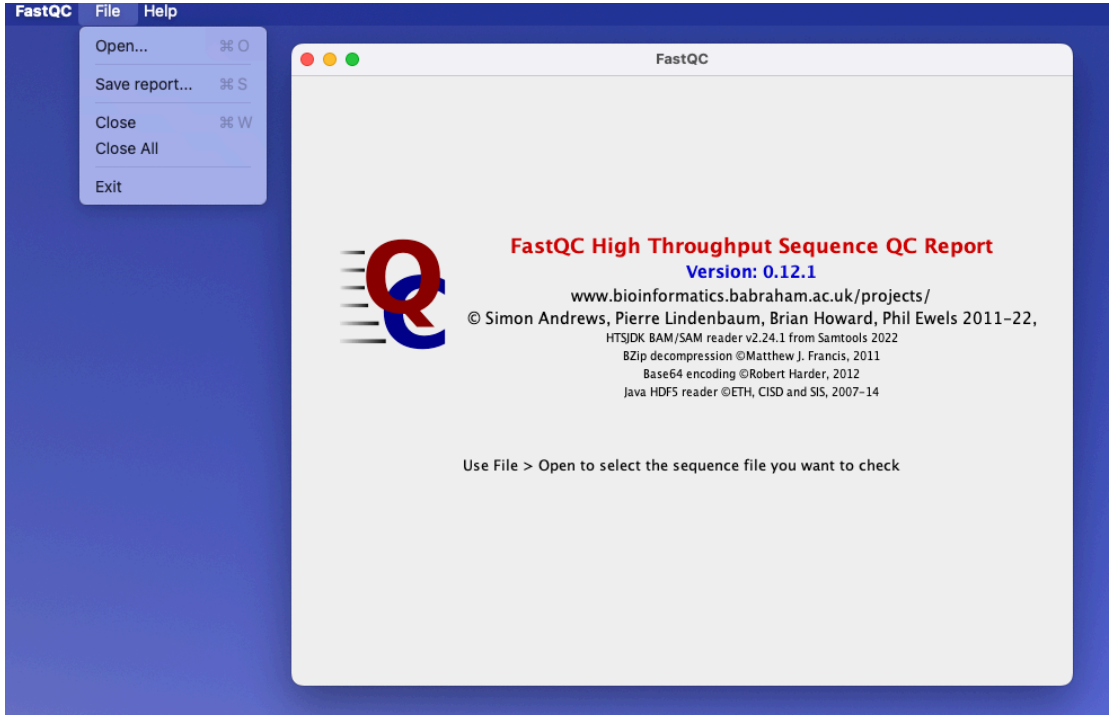
Example tool: FastQC

First step in RNASeq data analysis: QC of read files (e.g. *.fastq)



FastQC has a GUI





Are we FAIR, yet?

- where did I click
- reproducibility
- record exactly what I've done
- history
- instruction
- tool version
-

Command line tool

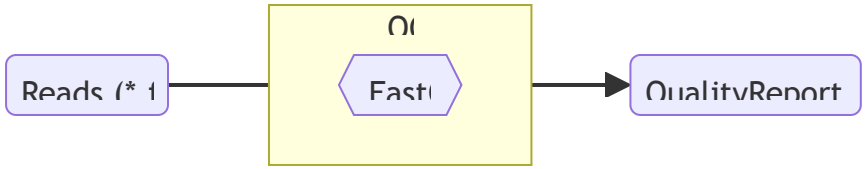
- Some tool that you can run ... on the command line
- Example:
  - CLI: **ARC Commander**
  - (GUI: **ARCitect**)
- Takes arguments or parameters as **inputs**
- Generates **outputs**

FastQC via command line

```
fastqc --version
fastqc --help
```

FastQC via command line

```
fastqc assays/rnaseq/dataset/blau1_CGATGT_L005_R1_002.fastq.gz
```



Materials & Methods ✅

```
fastqc assays/rnaseq/dataset/sample1
fastqc assays/rnaseq/dataset/sample2
fastqc assays/rnaseq/dataset/ ...
```

"*FastQC v0.12.1 was employed for read quality control using default parameters.*"

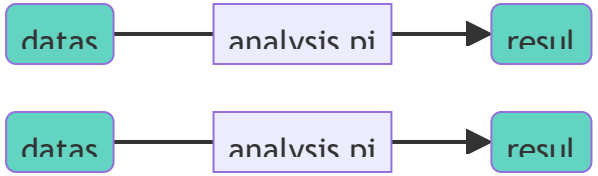
Installing bioinformatic tools 📈

- From source: https://www.bioinformatics.babraham.ac.uk/projects/download.html#fastqc
- Docker: `docker pull quay.io/biocontainers/fastqc`
- Conda: `conda install fastqc`

Why CWL and ARCs?

Reproducibility / Replicability of the data

Rerunning the same analysis on the **same** dataset



Reusability of the analysis

Applying the same analysis on **another** dataset

Some factors affecting reproducibility & reusability

- Version of tool / software
- Version of package/library and interpreter (python, R, F#, etc.)
- Operating system (linux, win, mac) and version
- ...

Approaches towards CWL in ARCs

1. Wrap a script
2. Wrap a CLI tool
3. Reuse an existing CWL document (command line tool or full workflow)
4. ...

Demo: CWL-Wrapping the CommandLineTool FastQC

Step 1

- Without in/out
- **Local tool installed**

```
#!/usr/bin/env cwl-runner
cwlVersion: v1.2
class: CommandLineTool

baseCommand: ["fastqc", "--help"]

inputs: []

outputs: []
```

Step 2: Add a docker container

```
#!/usr/bin/env cwl-runner
cwlVersion: v1.2
class: CommandLineTool

hints:
  DockerRequirement:
    dockerPull: quay.io/biocontainers/fastqc:0.11.9--hdfd78af_1

baseCommand: ["fastqc", "--help"]

inputs: []

outputs: []
```

Step 3: Define inputs

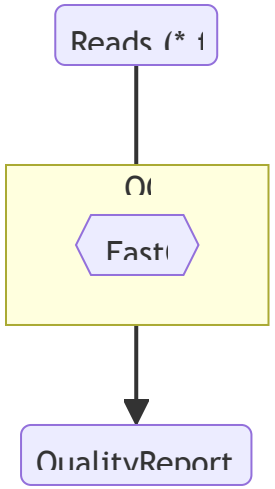
```
#!/usr/bin/env cwl-runner
cwlVersion: v1.2
class: CommandLineTool

hints:
  DockerRequirement:
    dockerPull: quay.io/biocontainers/fastqc:0.11.9--hdfd78af_1

baseCommand: ["fastqc"]

inputs:
  reads:
    type: File[]
    inputBinding:
      position: 1

arguments:
  - valueFrom: $(runtime.outdir)
    prefix: "-o"

outputs: []
```

Step 4: Define outputs

```
#!/usr/bin/env cwl-runner
cwlVersion: v1.2
class: CommandLineTool

hints:
  DockerRequirement:
    dockerPull: quay.io/biocontainers/fastqc:0.11.9--hdfd78af_1

baseCommand: ["fastqc"]

inputs:
  reads:
    type: File[]
    inputBinding:
      position: 1

arguments:
  - valueFrom: $(runtime.outdir)
    prefix: "-o"

outputs:
  fastqc_out:
      type: File[]
      outputBinding:
        glob:
          - "*_fastqc.zip"
          - "*_fastqc.html"
```

Step 4: Define outputs

```
#!/usr/bin/env cwl-runner
cwlVersion: v1.2
class: CommandLineTool

hints:
  DockerRequirement:
    dockerPull: quay.io/biocontainers/fastqc:0.11.9--hdfd78af_1

baseCommand: ["fastqc"]

inputs:
  reads:
    type: File[]
    inputBinding:
      position: 1

arguments:
  - valueFrom: $(runtime.outdir)
    prefix: "-o"

outputs:
  fastqc_out:
      type: File[]
      outputBinding:
        glob:
          - "*_fastqc.zip"
          - "*_fastqc.html"
```



Run the workflow

You can provide arguments via another file:

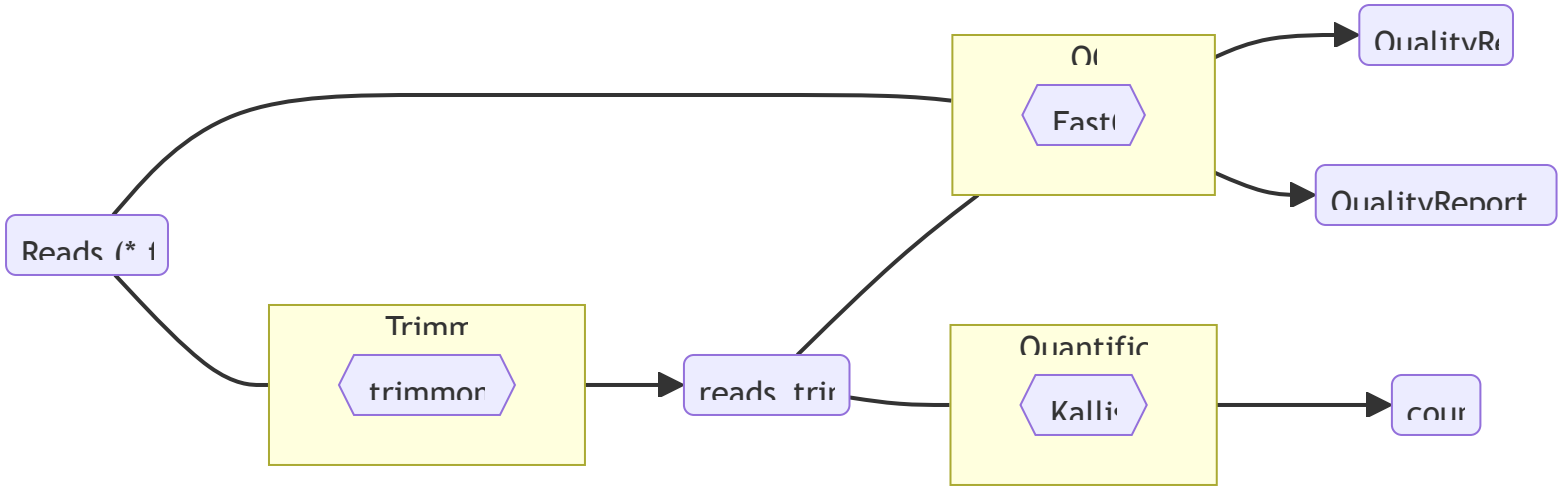
`run.yml`

```yaml
reads:
  - class: File
    path: ../../assays/rnaseq/dataset/blau1_CGATGT_L005_R1_002.fastq.gz
  - class: File
    path: ../../assays/rnaseq/dataset/blau2_TGACCA_L005_R1_002.fastq.gz
```

Reusability: Simply import an existing CWL

- e.g. from one ARC to another

Growing pipeline: First steps RNASeq pipeline



CWL is a time investment at first

There's a *tiny* learning curve and some dependencies

- JavaScript
- Docker
- Conda and the cwltool (reference runner)
- ...

...but it pays off!

ARC-CWL tutorials

- Knowledge Base: https://nfdi4plants.github.io/nfdi4plants.knowledgebase/guides/arc-cwl/

Resources

- Specification v1.2: https://www.commonwl.org/v1.2/CommandLineTool.html
- CWL tool: https://github.com/common-workflow-language/cwltool
- CWL Discourse: https://cwl.discourse.group
- Published Workflows: https://view.commonwl.org/workflows
- CWL repos: https://www.commonwl.org/repos/
- bio-cwl-tools: https://github.com/common-workflow-library/bio-cwl-tools/tree/release
- EBI-Metagenomics: https://github.com/EBI-Metagenomics/workflow-is-cwl/tree/master/tools