Nextflow DSL2 Cheat Sheet

Overview

Nextflow is a workflow management system for creating scalable and reproducible computational pipelines. It supports container technologies like Docker and Singularity for consistent execution environments.

Basic Syntax

Pipeline Structure

```
#!/usr/bin/env nextflow
nextflow.enable.dsl=2

process processName {
    input:
        // Define inputs (e.g., files, values)
    output:
        // Define outputs
    script:
        // Bash or other script commands
}
```

Running a Pipeline

```
nextflow run pipeline.nf
nextflow run pipeline.nf -resume  # Resume from cached results
nextflow run pipeline.nf -bg      # Run in background
nextflow run pipeline.nf -with-report report.html  # Generate execution report
```

Key Directives

Process Directives

```
process example {
    cpus 4                  // Allocate CPU cores
    memory '8 GB'           // Allocate memory
    time '2h'               // Set time limit
    queue 'long'            // Specify queue (for HPC)
    maxRetries 3            // Retry on failure
    errorStrategy 'retry'   // Options: 'terminate', 'finish', 'ignore', 'retry'
}
```

Workflow Directives

```
workflow {
    process1(input)
    process2(process1.out)
}
```

Containers

Using Docker

```
process dockerProcess {
    container 'biocontainers/fastqc:v0.11.9_cv8'
    input:
        path input_file
    output:
        path "output/*"
    script:
        """
        fastqc $input_file -o output
        """
}
```

- **Enable Docker**: Add to `nextflow.config`:

  ```
  docker.enabled = true
  ```

- **Run with Docker**:

  ```
  nextflow run pipeline.nf -with-docker
  ```

Using Singularity

```
process singularityProcess {
    container 'docker://biocontainers/fastqc:v0.11.9_cv8'
    input:
        path input_file
    output:
        path "output/*"
    script:
        """
        fastqc $input_file -o output
        """
}
```

- **Enable Singularity**: Add to `nextflow.config`:

  ```
  singularity.enabled = true
  singularity.autoMounts = true
  ```

- **Run with Singularity**:

  ```
  nextflow run pipeline.nf -with-singularity
  ```

Container Options

- **Docker Options**:

  ```
  docker.runOptions = '-u $(id -u):$(id -g)'
  ```

- **Singularity Options**:

  ```
  singularity.runOptions = '--bind /data:/data'
  ```

Channels

Creating Channels

```
Channel.fromPath('data/*.fastq')  // From files
Channel.fromList([1, 2, 3])       // From a list
Channel.of(1, 2, 3)               // From values
Channel.fromFilePairs('data/*_{1,2}.fastq')  // Paired-end files
```

Channel Operations

```
input_ch.view()                   // View channel contents
input_ch.map { it -> it*2 }       // Transform elements
input_ch.filter { it > 5 }        // Filter elements
input_ch.groupTuple()             // Group by key
```

Configuration

`nextflow.config`

```
process {
    executor = 'slurm'            // Options: 'local', 'sge', 'lsf', 'slurm'
    queue = 'production'
    cpus = 2
    memory = '4 GB'
}

docker {
    enabled = true
    temp = 'auto'
}

singularity {
    enabled = true
    autoMounts = true
}
```

Common Commands

Pipeline Execution

```
nextflow run pipeline.nf -c config_file.config  # Custom config
nextflow clean -f                       # Clean cache
nextflow info                           # Show Nextflow info
nextflow list                           # List pipelines
```

Debugging

```
nextflow run pipeline.nf -with-trace trace.txt  # Generate trace file
nextflow run pipeline.nf -with-timeline timeline.html  # Execution timeline
```

Tips

- **Caching**: Use `-resume` to reuse cached results.
- **Modules**: Use `include { processName } from './module.nf'` for modularity.
- **Parameters**: Define in `nextflow.config` or pass via `--param value`.

- **Container Best Practices**:
  - Use specific container tags (e.g., `v0.11.9_cv8`).
  - Test containers locally before pipeline execution.
  - Ensure container compatibility with input/output paths.

Resources

- Nextflow Documentation
- Docker Hub
- Singularity Hub