

Workflows and Operators in nextflow








**Pre-training
nf-core hackathon 2025**

March 10th 2025

Antoine Buetti-Dinh
antoine.buetti@gmail.com

Highlights of Today

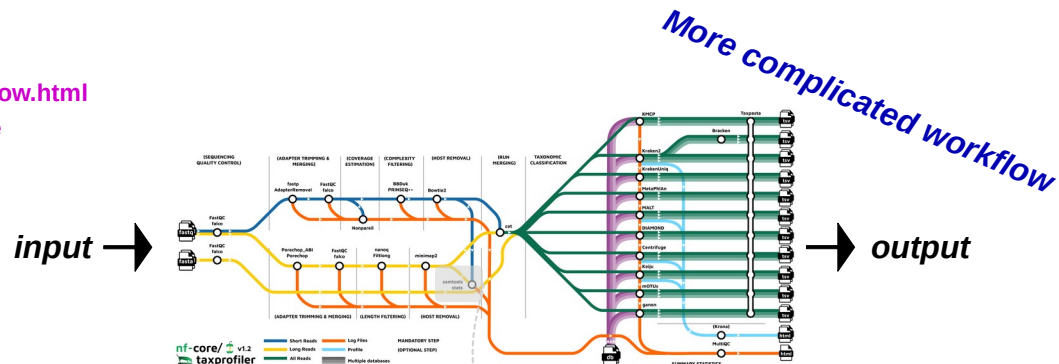
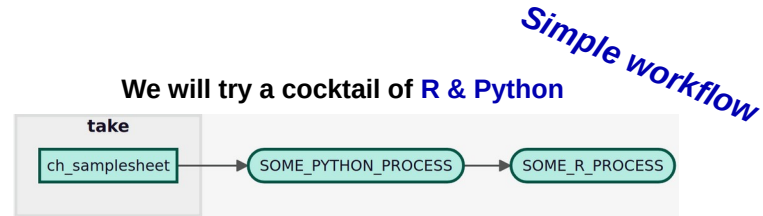
1. Workflows in nextflow

- Workflows (pipelines) of **multiple steps** that perform analyses in data science
- Example of a **mix** of programming languages
 - Python, R, Bash in **same script**
 - **Transportable & reproducible** via **containers**  docker  CONDA®  Wave ...
 - Easy **deployment** on different infrastructure (servers, cloud, clusters)   aws  Google Cloud  Azure ...
- Refs:

<https://carpentries-incubator.github.io/workflows-nextflow/06-workflow.html>

https://training.nextflow.io/latest/basic_training/intro/#nextflow-code

https://training.nextflow.io/latest/basic_training/rnaseq_pipeline/



Include old software!
(by creating right environment)

2. Operators in nextflow

- Simple **manipulation** tools
 - Filter, gather, split, sort,...

- Refs:
 - <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>
 - https://training.nextflow.io/latest/basic_training/operators/

Workflow Example

Want a more bioinformatics example?

→ <https://carpentries-incubator.github.io/workflows-nextflow/06-workflow.html>

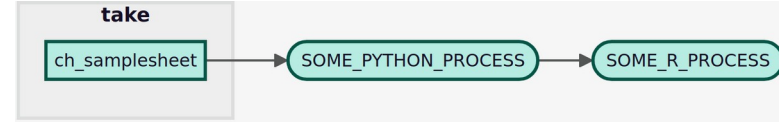
Workflow Example

Want a more bioinformatics example?

→ <https://carpentries-incubator.github.io/workflows-nextflow/06-workflow.html>

Just a file with a number in it

```
workflow {  
  input_ch = Channel.fromPath("input.csv")  
  SOME_PYTHON_PROCESS(input_ch)  
  SOME_R_PROCESS(SOME_PYTHON_PROCESS.out)  
}
```



```
process SOME_PYTHON_PROCESS {  
  input:  
  path input_file  
  
  output:  
  path *_processed.txt  
  
  script:  
  """  
  #!/usr/bin/env python3  
  
  import sys  
  import random  
  
  with open("${input_file}", "r") as f:  
    number = int(f.read().strip())  
  
  result = [random.randint(5, 15) for i in range(0, number)]  
  
  with open("${input_file.baseName}_processed.txt", "w") as f:  
    for item in result:  
      f.write(f"{item}\n")  
  """  
}
```

```
process SOME_R_PROCESS {  
  input:  
  path input_file  
  
  output:  
  path *_analyzed.txt  
  
  script:  
  """  
  #!/usr/bin/env Rscript  
  
  input_data <- read.table("${input_file}", header=FALSE,  
    col.names=c("value"))  
  summary_stats <- summary(input_data$value)  
  
  sink("${input_file.baseName}_analyzed.txt")  
  cat("Summary Statistics:\n")  
  print(summary_stats)  
  
  sink()  
  """  
}
```

Generates random numbers

11
6
15
5
15
14
.
.
.

Calculates Summary statistics

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
5.00	6.25	11.50	10.14	13.00	15.00

Workflow Example

Want a more bioinformatics example?

→ <https://carpentries-incubator.github.io/workflows-nextflow/06-workflow.html>

Just a file with a number in it

```
workflow {  
  input_ch = Channel.fromPath("input.csv")  
  SOME_PYTHON_PROCESS(input_ch)  
  SOME_R_PROCESS(SOME_PYTHON_PROCESS.out)  
}
```



```
process SOME_PYTHON_PROCESS {  
  container 'python:3.9'  
  input:  
    path input_file  
  
  output:  
    path '*_processed.txt'  
  
  script:  
    """  
    #!/usr/bin/env python3  
  
    import sys  
    import random  
  
    with open("${input_file}", "r") as f:  
      number = int(f.read().strip())  
  
    result = [random.randint(5, 15) for i in range(0, number)]  
  
    with open("${input_file.baseName}_processed.txt", "w") as f:  
      for item in result:  
        f.write(f"{item}\n")  
    """  
}
```

You can ask to use particular versions/packages via containers (e.g. Docker or Conda)

```
process SOME_R_PROCESS {  
  conda 'r-base=4.1.0'  
  input:  
    path input_file  
  
  output:  
    path '*_analyzed.txt'  
  
  script:  
    """  
    #!/usr/bin/env Rscript  
  
    input_data <- read.table("${input_file}", header=FALSE,  
    col.names=c("value"))  
    summary_stats <- summary(input_data$value)  
  
    sink("${input_file.baseName}_analyzed.txt")  
    cat("Summary Statistics:\n")  
    print(summary_stats)  
  
    sink()  
    """  
}
```

Generates random numbers

11
6
15
5
15
14
.
.
.

Calculates Summary statistics

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
5.00	6.25	11.50	10.14	13.00	15.00

Let's see some more cool things one can do!



Another Example...

Another Example...

Also something quite simpler...

```
├─ parallel.nf
├─ sample1.fastq.gz
├─ sample2.fastq.gz
├─ sample3.fastq.gz
├─ sample4.fastq.gz
├─ sample5.fastq.gz
└─ sample6.fastq.gz
```

```
1  #!/usr/bin/env nextflow
2
3  process myProcess {
4      input:
5          path myInp
6
7      output:
8          path '*.fastq'
9
10     script:
11         """
12         gunzip -c $myInp > "${myInp.baseName}.fastq"
13         """
14     }
15
16     workflow {
17         myChannel = Channel.fromPath("./*.fastq.gz")
18         myProcess(myChannel)
19     }
20
```

My simple process to decompress 6 zip files

Simple call from main

Another Example...

~ Concept of "channel"

Also something quite simpler...

```
├─ parallel.nf
├─ sample1.fastq.gz
├─ sample2.fastq.gz
├─ sample3.fastq.gz
├─ sample4.fastq.gz
├─ sample5.fastq.gz
└─ sample6.fastq.gz
```

```
1  #!/usr/bin/env nextflow
2
3  process myProcess {
4    input:
5      path myInp
6
7    output:
8      path '*.fastq'
9
10   script:
11     """
12     gunzip -c $myInp > "${myInp.baseName}.fastq"
13     """
14   }
15
16   workflow {
17     myChannel = Channel.fromPath("./*.fastq.gz")
18     myProcess(myChannel)
19   }
20
```

→ Run:

6 processes to run are spotted!

```
executor > local (6)
[4b/3a266b] myProcess (6) | 6 of 6 ✓
```

```
├─ parallel.nf
├─ sample1.fastq.gz
├─ sample2.fastq.gz
├─ sample3.fastq.gz
├─ sample4.fastq.gz
├─ sample5.fastq.gz
└─ sample6.fastq.gz
├─ work
├─ 08
├─   └─ b6ae829662ebf47836fbf7223f08c2
├─     └─ sample5.fastq.fastq
├─       └─ sample5.fastq.gz -> /home/antoinebueti/Desktop/tmp/parallel/sample5.fastq.gz
├─ 22
├─   └─ 893f66cfb7c7af3deedccc8271382f
├─     └─ sample4.fastq.fastq
├─       └─ sample4.fastq.gz -> /home/antoinebueti/Desktop/tmp/parallel/sample4.fastq.gz
├─ 2d
├─   └─ 740a9ce945adacb0dd012c544c310d
├─     └─ sample3.fastq.fastq
├─       └─ sample3.fastq.gz -> /home/antoinebueti/Desktop/tmp/parallel/sample3.fastq.gz
├─ 4b
├─   └─ 3a266ba49b13d36645494c0717376b
├─     └─ sample2.fastq.fastq
├─       └─ sample2.fastq.gz -> /home/antoinebueti/Desktop/tmp/parallel/sample2.fastq.gz
├─ b0
├─   └─ 58e4c8308156332c41e83c01e8432e
├─     └─ sample1.fastq.fastq
├─       └─ sample1.fastq.gz -> /home/antoinebueti/Desktop/tmp/parallel/sample1.fastq.gz
├─ d7
├─   └─ 0c9ac81adaf588748b42234660b6ff
├─     └─ sample6.fastq.fastq
├─       └─ sample6.fastq.gz -> /home/antoinebueti/Desktop/tmp/parallel/sample6.fastq.gz
```

Another Example...

Also something quite simpler...

```
parallel.nf
sample1.fastq.gz
sample2.fastq.gz
sample3.fastq.gz
sample4.fastq.gz
sample5.fastq.gz
sample6.fastq.gz
```

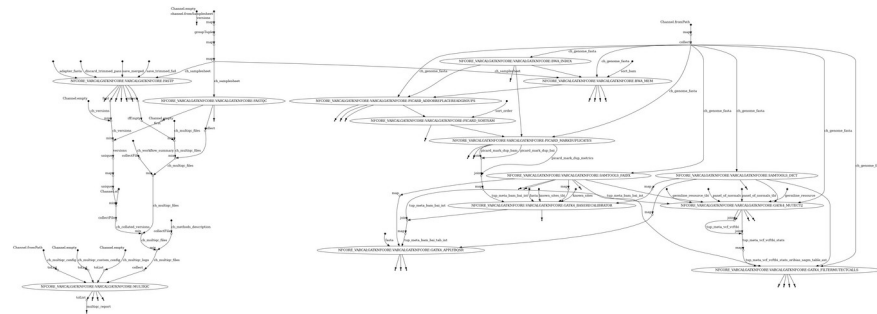
```
1  #!/usr/bin/env nextflow
2
3  process myProcess {
4    input:
5    path myInp
6
7    output:
8    path '*.fastq'
9
10   script:
11   """
12   gunzip -c $myInp > "${myInp.baseName}.fastq"
13   """
14 }
15
16 workflow {
17   myChannel = Channel.fromPath("./*.fastq.gz")
18   myProcess(myChannel)
19 }
20
```

→ Run:

6 processes to run are spotted!

```
executor > local (6)
[4b/3a266b] myProcess (6) | 6 of 6 ✓
```

Asynchronous programming



Branched **DAGs** allow to establish priorities

- Things to compute **get propagated** through the process structure
- According to **available resources**

Another Example...

Also something quite simpler...

```
parallel.nf
sample1.fastq.gz
sample2.fastq.gz
sample3.fastq.gz
sample4.fastq.gz
sample5.fastq.gz
sample6.fastq.gz

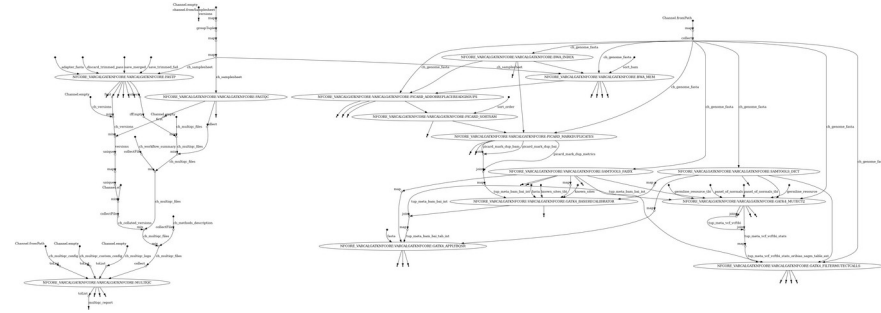
1  #!/usr/bin/env nextflow
2
3  process myProcess {
4    input:
5    path myInp
6
7    output:
8    path '*.fastq'
9
10   script:
11   """
12   gunzip -c $myInp > "${myInp.baseName}.fastq"
13   """
14 }
15
16 workflow {
17   myChannel = Channel.fromPath("./*.fastq.gz")
18   myProcess(myChannel)
19 }
20
```

→ Run:

6 processes to run are spotted!

```
executor > local (6)
[4b/3a266b] myProcess (6) | 6 of 6 ✓
```

Asynchronous programming



Some other sequential code in R/Python/...

→ gets also **handled the parallel way!**

Branched **DAGs** allow to establish priorities

- Things to compute **get propagated** through the process structure
- According to **available resources**

That's why  nextflow is a good choice also for simple things 😊

Operators

Overview on some operators

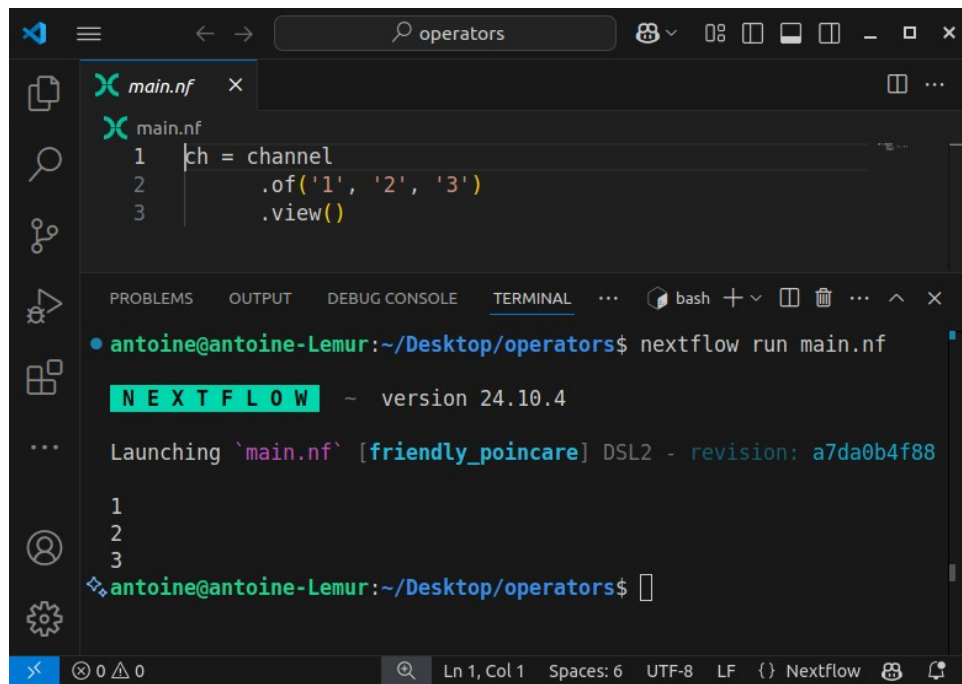
→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

That's why  nextflow is a good choice also for simple things 😊

Operators

Overview on some operators

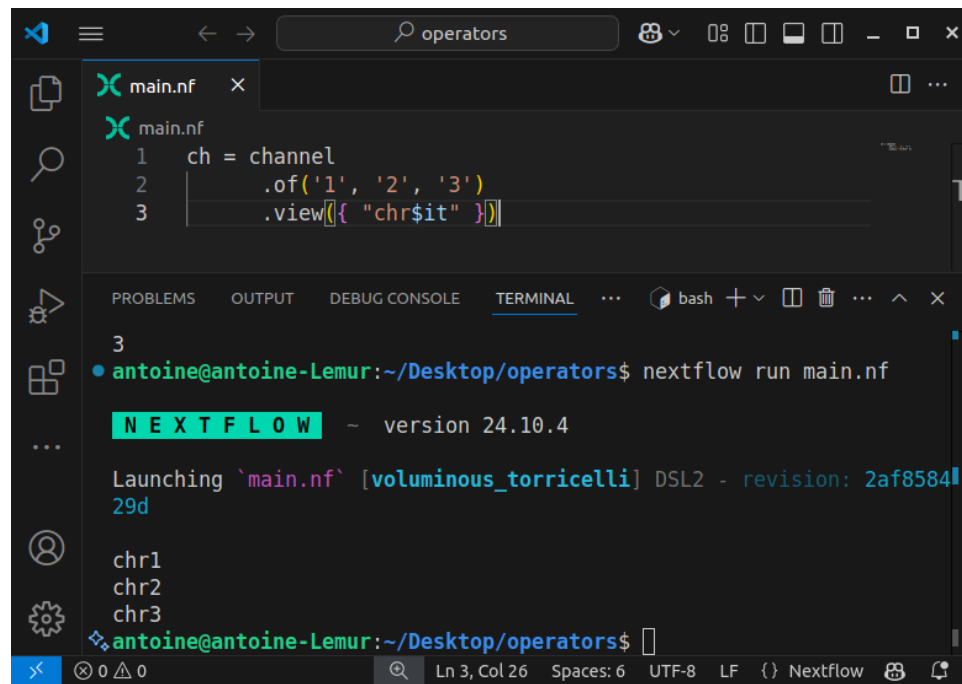
→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>



The screenshot shows a VS Code editor with a file named `main.nf`. The script contains three lines of code:

```
1 ch = channel
2   .of('1', '2', '3')
3   .view()
```

Below the editor, the terminal shows the command `nextflow run main.nf` being executed. The output indicates that Nextflow version 24.10.4 is running and that the DSL2 revision is `a7da0b4f88`. The output shows the numbers 1, 2, and 3, corresponding to the lines in the script.



The screenshot shows a VS Code editor with a file named `main.nf`. The script contains three lines of code:

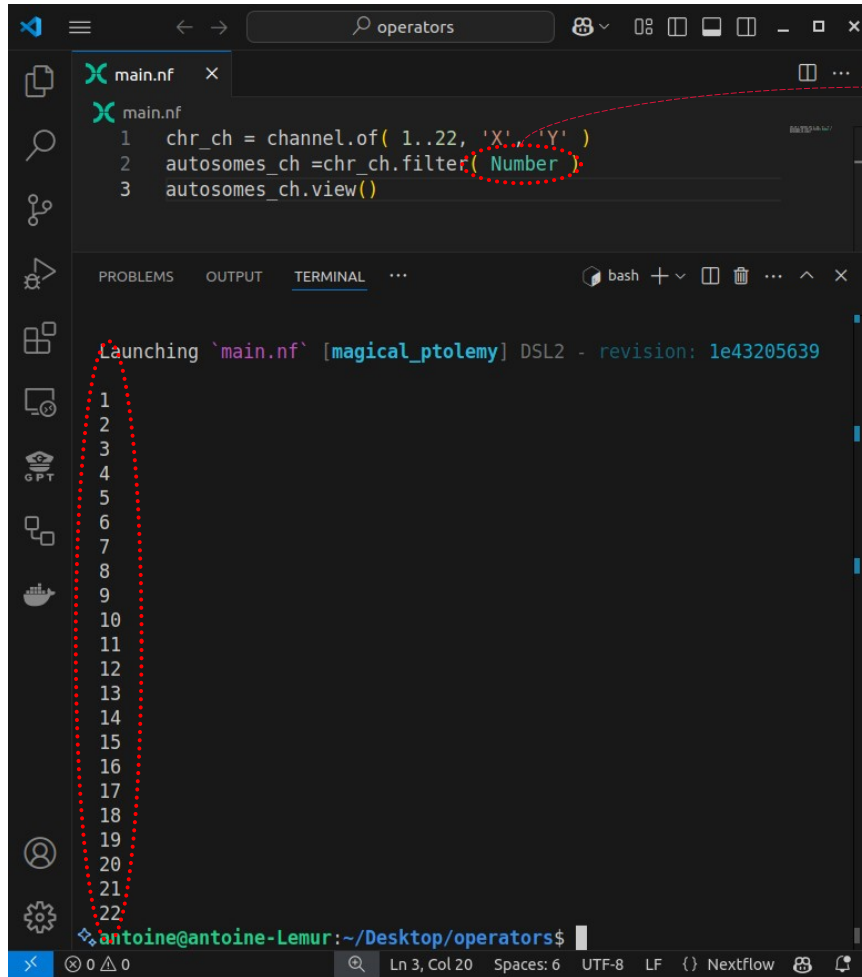
```
1 ch = channel
2   .of('1', '2', '3')
3   .view({ "chr${it}" })
```

Below the editor, the terminal shows the command `nextflow run main.nf` being executed. The output indicates that Nextflow version 24.10.4 is running and that the DSL2 revision is `2af8584`. The output shows the strings `chr1`, `chr2`, and `chr3`, corresponding to the lines in the script.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

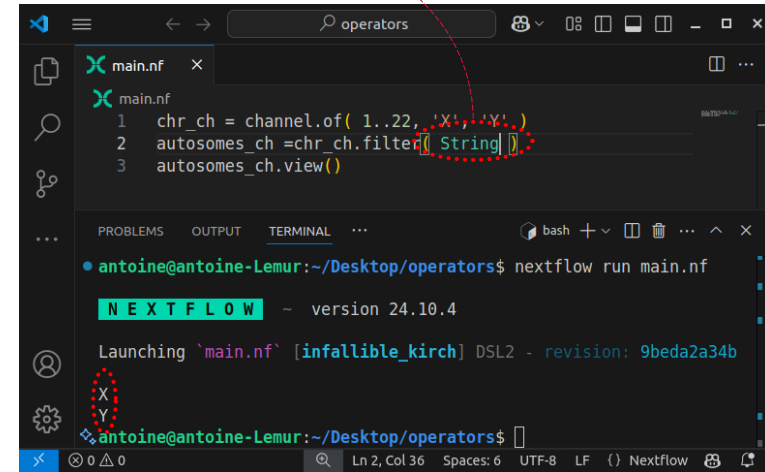


The screenshot shows a VS Code editor with a file named `main.nf`. The script contains three lines of DSL2 code:

```
1 chr_ch = channel.of( 1..22, 'X', 'Y' )
2 autosomes_ch = chr_ch.filter( Number )
3 autosomes_ch.view()
```

The word `Number` in line 2 is circled in red. Below the editor, the terminal window shows the command `nextflow run main.nf` being executed. The output indicates that the workflow was launched successfully, with the revision `1e43205639`. A red dashed line connects the `Number` operator in the script to the terminal output.

```
Launching 'main.nf' [magical_ptolemy] DSL2 - revision: 1e43205639
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
antoine@antoine-Lemur:~/Desktop/operators$
```



The screenshot shows a VS Code editor with a file named `main.nf`. The script contains three lines of DSL2 code:

```
1 chr_ch = channel.of( 1..22, 'X', 'Y' )
2 autosomes_ch = chr_ch.filter( String )
3 autosomes_ch.view()
```

The word `String` in line 2 is circled in red. Below the editor, the terminal window shows the command `nextflow run main.nf` being executed. The output indicates that the workflow was launched successfully, with the revision `9beda2a34b`. A red dashed line connects the `String` operator in the script to the terminal output.

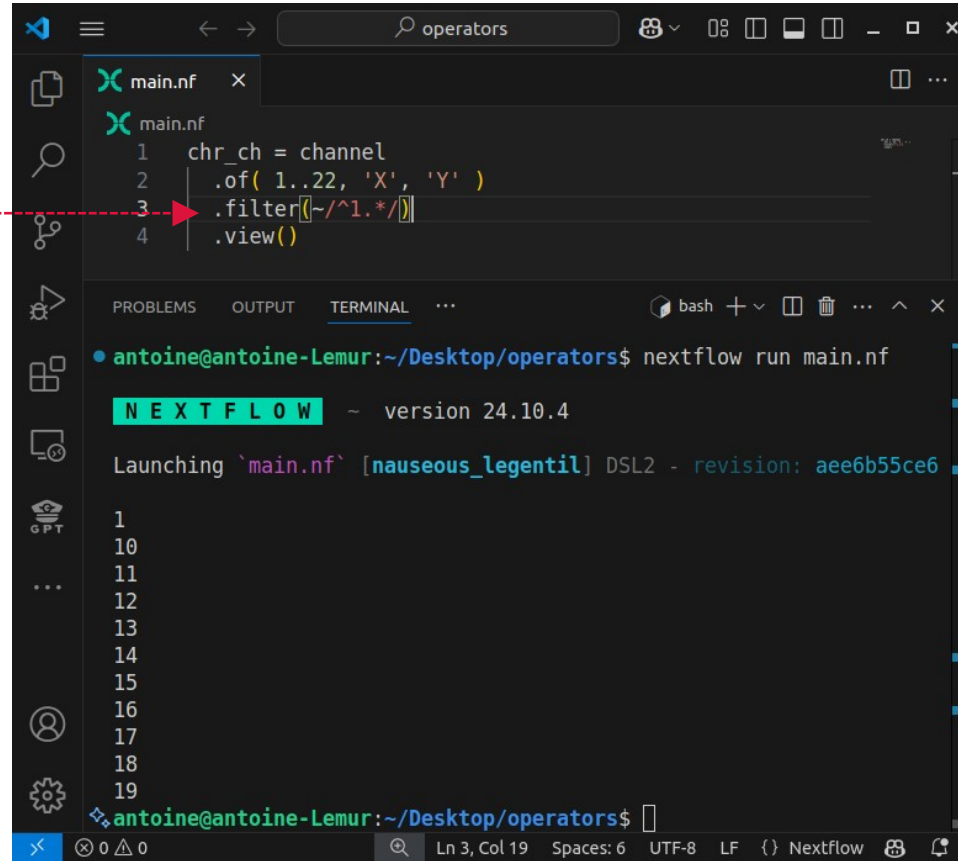
```
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf
NEXTFLOW ~ version 24.10.4
Launching 'main.nf' [infallible_kirch] DSL2 - revision: 9beda2a34b
antoine@antoine-Lemur:~/Desktop/operators$
```

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Regular expressions !



The screenshot shows a Nextflow workflow editor with a file named `main.nf`. The workflow contains the following code:

```
1 chr_ch = channel
2   .of( 1..22, 'X', 'Y' )
3   .filter[~/^1.*~/]
4   .view()
```

A red dashed arrow points from the text "Regular expressions !" to the regular expression `~/^1.*~/` in the `.filter` operator on line 3.

The terminal output shows the following:

```
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [nauseous_legentil] DSL2 - revision: aee6b55ce6

1
10
11
12
13
14
15
16
17
18
19

antoine@antoine-Lemur:~/Desktop/operators$
```

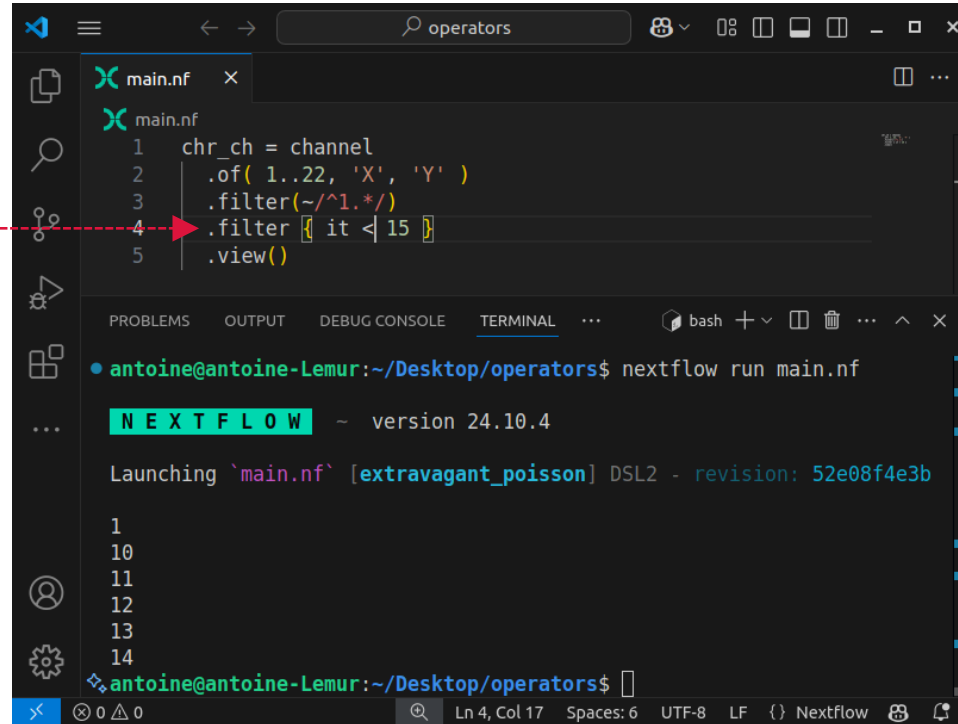
The status bar at the bottom indicates the cursor is at line 3, column 19, with 6 spaces, in UTF-8 encoding, using LF line endings, and the file is edited with Nextflow.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Filtering !



The screenshot shows a Nextflow workflow editor with a file named `main.nf`. The workflow code is as follows:

```
1 chr_ch = channel
2   .of( 1..22, 'X', 'Y' )
3   .filter(~/^1.*/ )
4   .filter { it < 15 }
5   .view()
```

A red dashed arrow points from the text "Filtering !" to the `.filter { it < 15 }` line in the code. The editor interface includes a search bar at the top with the text "operators", a sidebar with icons for file management, search, and settings, and a terminal at the bottom. The terminal shows the command `nextflow run main.nf` being executed, followed by the output:

```
NEXTFLOW ~ version 24.10.4
Launching `main.nf` [extravagant_poisson] DSL2 - revision: 52e08f4e3b
1
10
11
12
13
14
```

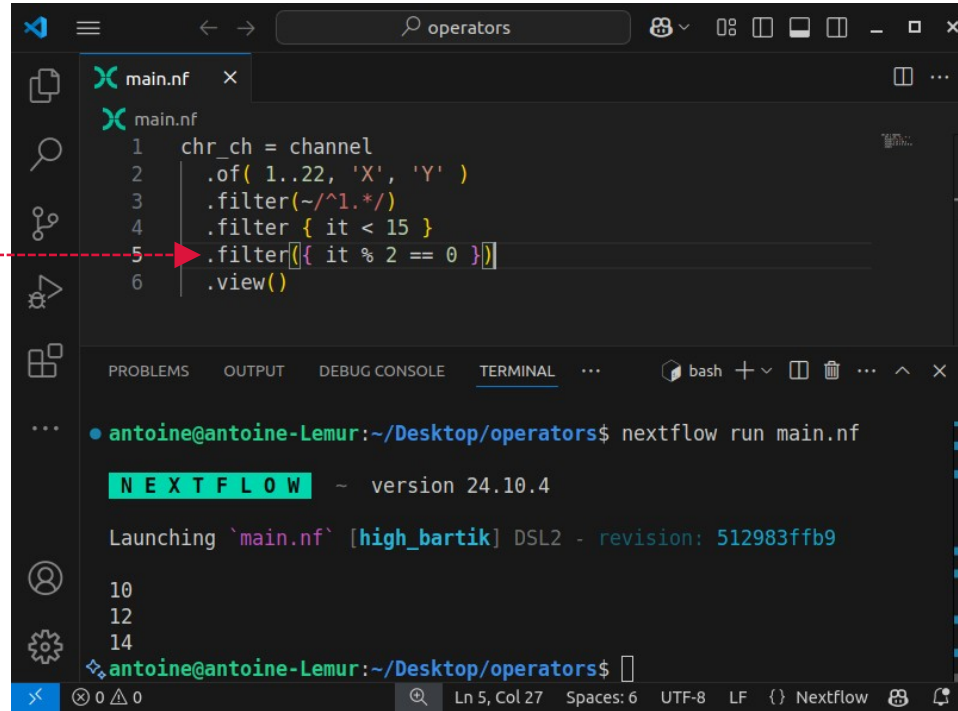
The terminal output shows the first 14 elements of the channel, which are integers from 1 to 14, indicating that the filtering operation successfully removed the 'X' and 'Y' elements.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Filtering !



The screenshot shows a Nextflow workflow editor with a file named `main.nf`. The workflow code is as follows:

```
1 chr_ch = channel
2   .of( 1..22, 'X', 'Y' )
3   .filter(~/^1.*/ )
4   .filter { it < 15 }
5   .filter({ it % 2 == 0 })
6   .view()
```

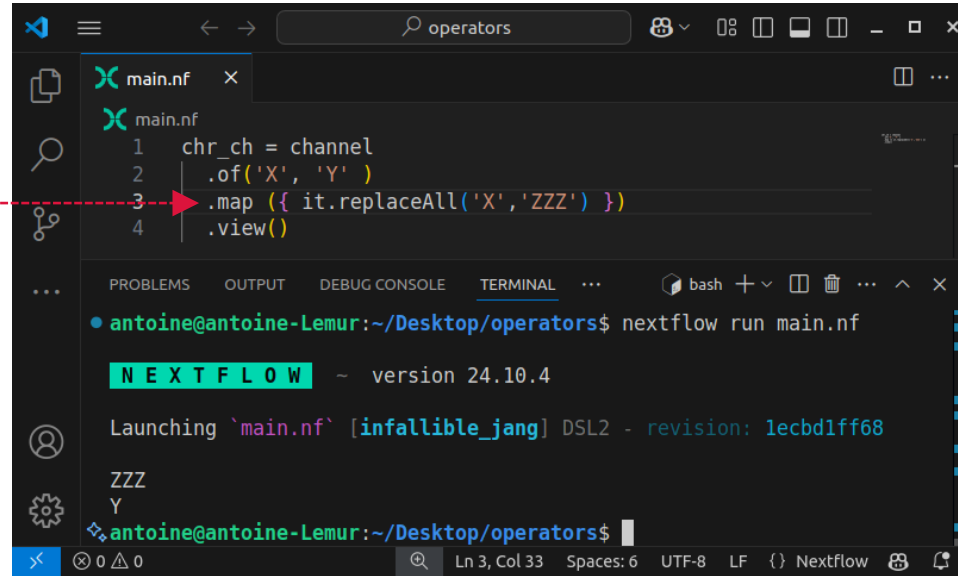
A red dashed line points from the text "Filtering !" to the `.filter({ it % 2 == 0 })` line in the code. The terminal at the bottom shows the command `nextflow run main.nf` being executed, displaying the Nextflow version (24.10.4) and the DSL2 revision (512983ffb9). The output of the workflow is shown as a list of numbers: 10, 12, 14.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Replace patterns !



The screenshot shows a VS Code editor window with a file named `main.nf`. The script contains the following code:

```
1 chr_ch = channel
2   .of('X', 'Y' )
3   .map ({ it.replaceAll('X','ZZZ') })
4   .view()
```

A red dashed line points from the text "Replace patterns !" to the `replaceAll` method in line 3. Below the editor, the terminal shows the output of running `nextflow run main.nf`:

```
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [infallible_jang] DSL2 - revision: 1ecbd1ff68

ZZZ
Y
antoine@antoine-Lemur:~/Desktop/operators$
```

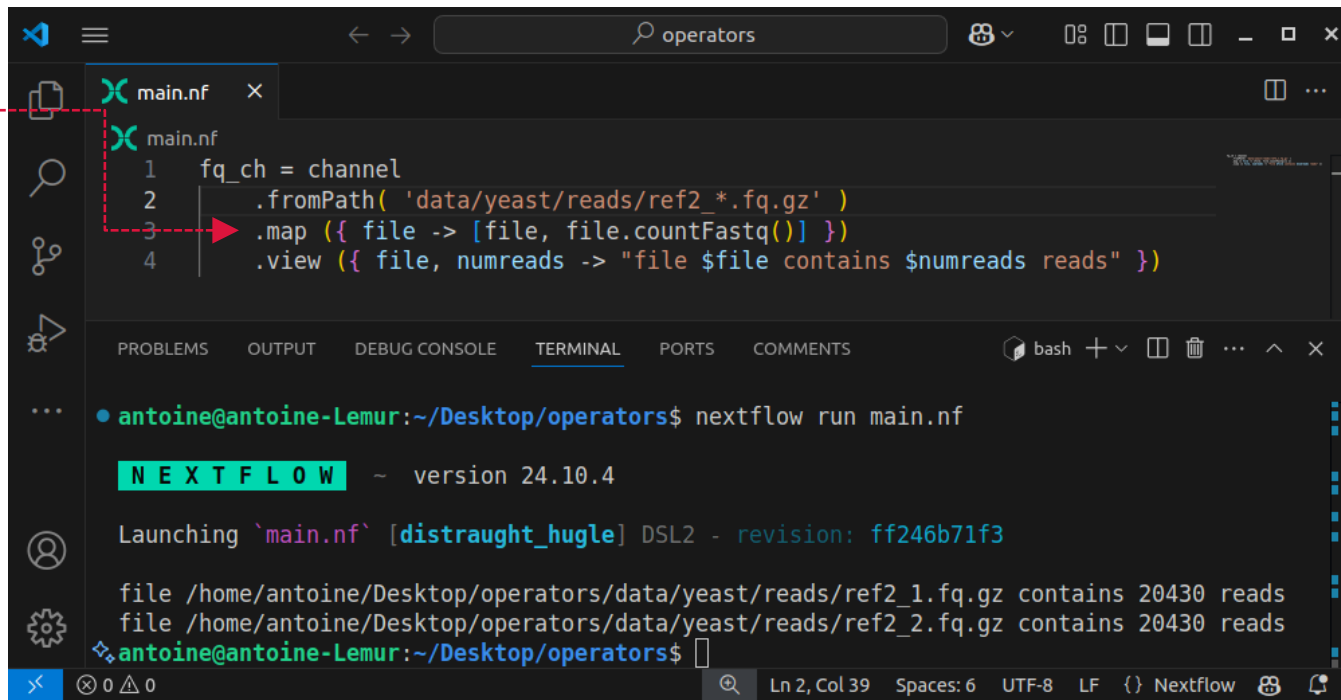
The status bar at the bottom indicates the cursor is at line 3, column 33, with 6 spaces, in UTF-8 encoding, using LF line endings, and the file is a Nextflow script.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Somewhat specific tools
(fastq stuff)
can be integrated



The screenshot shows a Nextflow workflow editor with a dark theme. The top toolbar includes icons for file operations, search, and window management. The main editor displays a file named `main.nf` with the following code:

```
1 fq_ch = channel
2   .fromPath( 'data/yeast/reads/ref2_*.fq.gz' )
3   .map ( { file -> [file, file.countFastq()] } )
4   .view ( { file, numreads -> "file $file contains $numreads reads" } )
```

A red dashed box highlights the first three lines of code, and a red arrow points to the `.map` operator on line 3. Below the editor, the `TERMINAL` tab is active, showing the execution of the workflow:

```
• antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf
NEXTFLOW ~ version 24.10.4
Launching `main.nf` [distraught_hugle] DSL2 - revision: ff246b71f3
file /home/antoine/Desktop/operators/data/yeast/reads/ref2_1.fq.gz contains 20430 reads
file /home/antoine/Desktop/operators/data/yeast/reads/ref2_2.fq.gz contains 20430 reads
• antoine@antoine-Lemur:~/Desktop/operators$
```

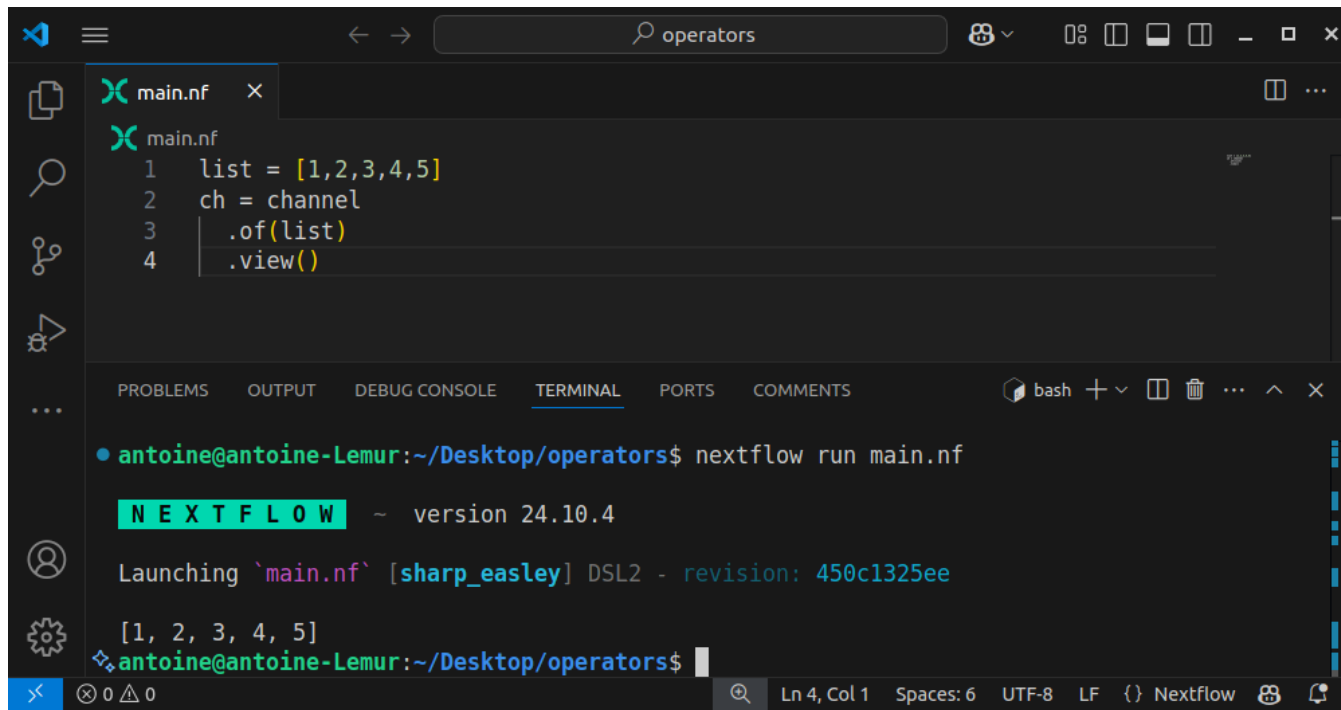
The status bar at the bottom indicates the current position is `Ln 2, Col 39` with `Spaces: 6`, `UTF-8` encoding, and `LF` line endings. The `Nextflow` logo is also visible in the status bar.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Values vs. lists
Flatten vs. collect



The screenshot shows a VS Code editor with a file named `main.nf` open. The script contains the following code:

```
1 list = [1,2,3,4,5]
2 ch = channel
3   .of(list)
4   .view()
```

Below the editor, the `TERMINAL` panel is active, showing the output of the command `nextflow run main.nf`. The output includes the Nextflow version (24.10.4) and the DSL2 revision (450c1325ee). The final output of the script is the list `[1, 2, 3, 4, 5]`.

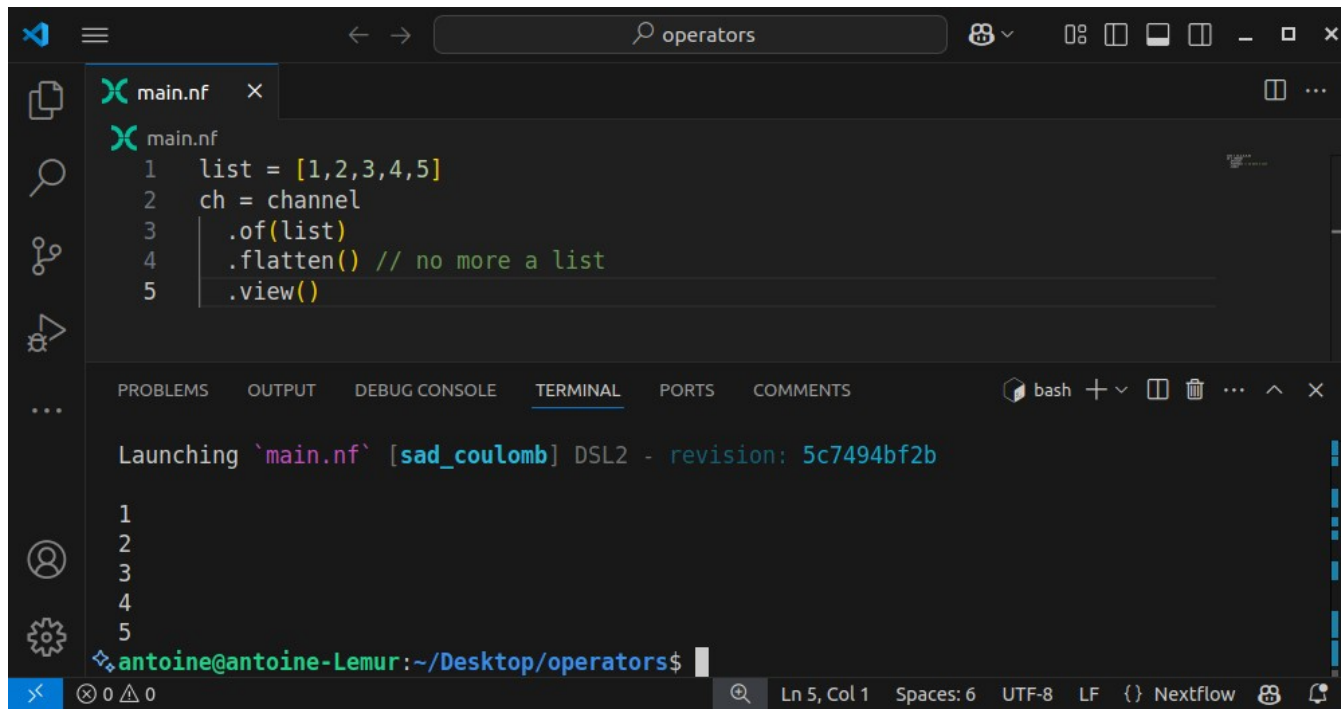
```
• antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf
NEXTFLOW ~ version 24.10.4
Launching `main.nf` [sharp_easley] DSL2 - revision: 450c1325ee
[1, 2, 3, 4, 5]
❖ antoine@antoine-Lemur:~/Desktop/operators$
```

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Values vs. lists
Flatten vs. collect



The screenshot shows a Nextflow IDE interface. The top panel displays a workflow script named `main.nf` with the following code:

```
1 list = [1,2,3,4,5]
2 ch = channel
3   .of(list)
4   .flatten() // no more a list
5   .view()
```

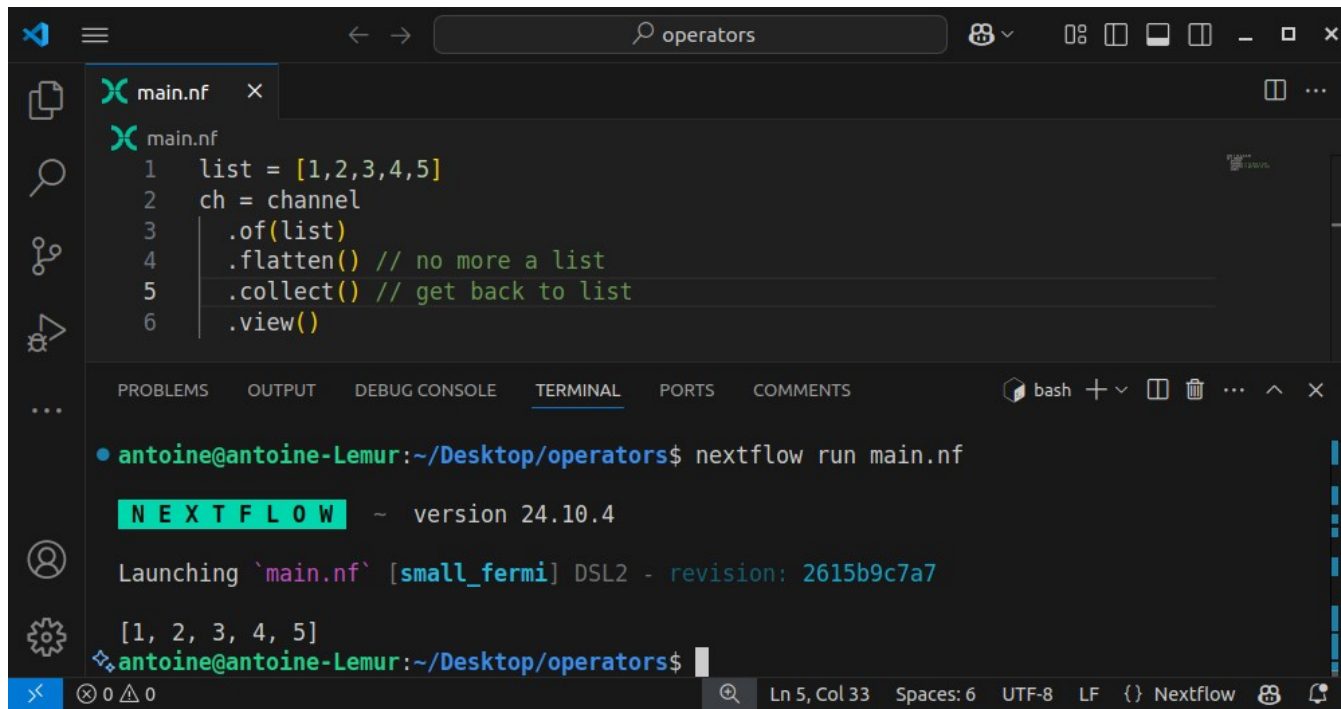
The bottom panel shows the execution output in the `TERMINAL` tab. It indicates that the workflow `'main.nf'` was launched using the `sad_coulomb` DSL2 engine with revision `5c7494bf2b`. The output shows a list of numbers 1 through 5, followed by a prompt to run the command `antoin@antoin-Lemur:~/Desktop/operators$`.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Values vs. lists
Flatten vs. collect



The screenshot shows a Nextflow IDE interface. The top panel displays a script named `main.nf` with the following content:

```
1 list = [1,2,3,4,5]
2 ch = channel
3     .of(list)
4     .flatten() // no more a list
5     .collect() // get back to list
6     .view()
```

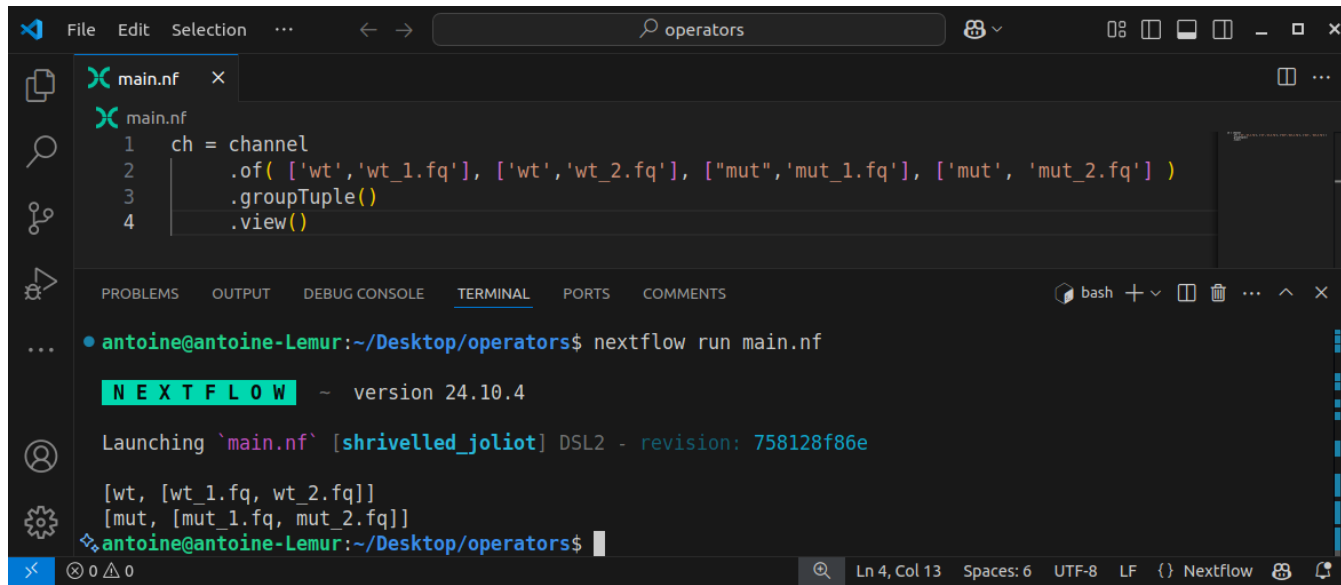
The bottom panel shows the terminal output after running the script with `nextflow run main.nf`. The output indicates the Nextflow version (24.10.4) and the DSL2 revision (2615b9c7a7). The final output is a list: `[1, 2, 3, 4, 5]`.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Group stuff together!
Such as read pairs



The screenshot shows a VS Code editor with a file named `main.nf` open. The script contains the following code:

```
1 ch = channel
2   .of( ['wt','wt_1.fq'], ['wt','wt_2.fq'], ["mut",'mut_1.fq'], ['mut', 'mut_2.fq'] )
3   .groupTuple()
4   .view()
```

Below the editor, the terminal window shows the output of running `nextflow run main.nf`. The output includes the Nextflow version (24.10.4) and the DSL2 revision (758128f86e). The script successfully executes, displaying the grouped read pairs as two lists: `[wt, [wt_1.fq, wt_2.fq]]` and `[mut, [mut_1.fq, mut_2.fq]]`.

```
• antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

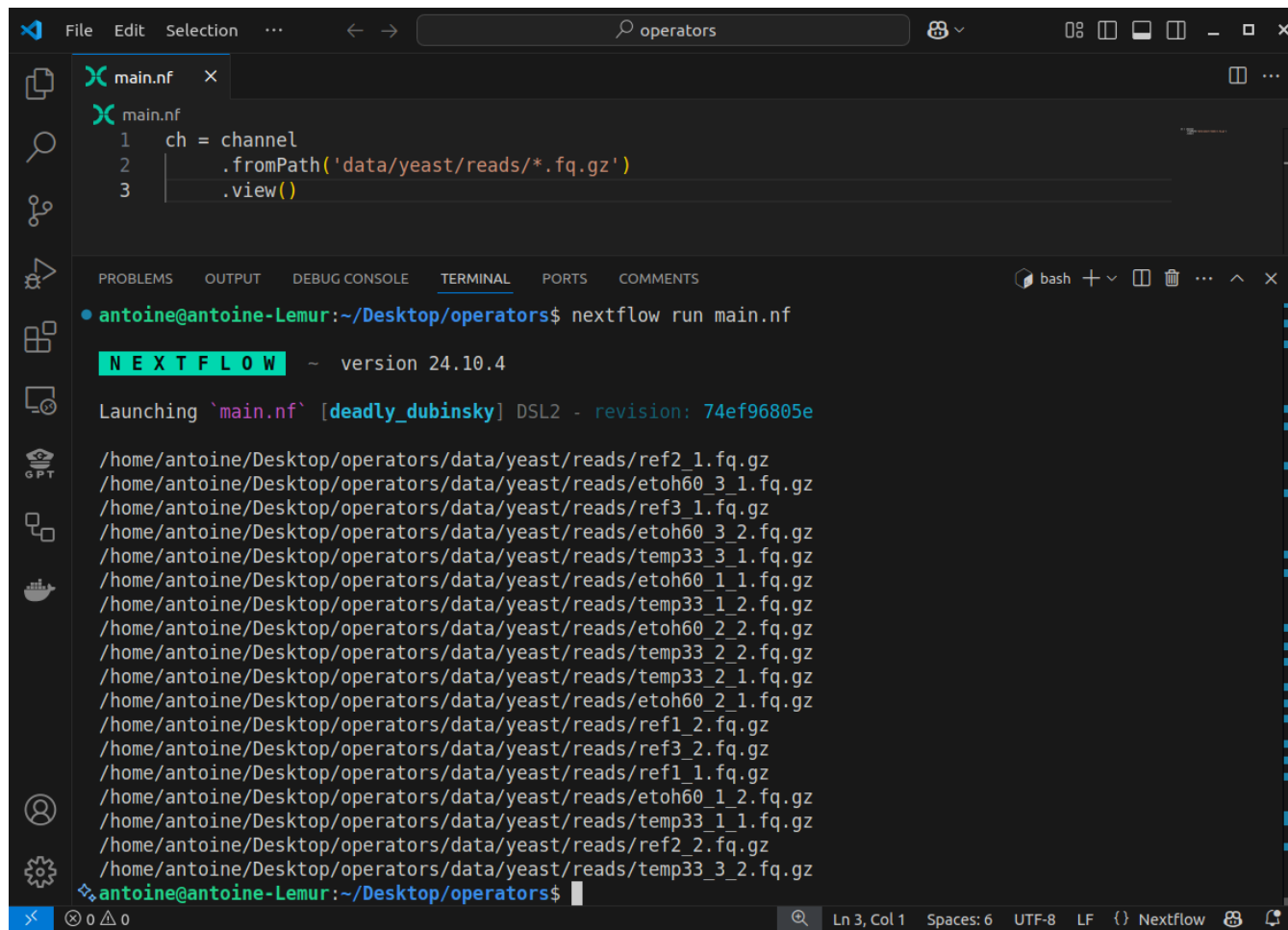
Launching `main.nf` [shrivelled_joliot] DSL2 - revision: 758128f86e

[wt, [wt_1.fq, wt_2.fq]]
[mut, [mut_1.fq, mut_2.fq]]
♦ antoine@antoine-Lemur:~/Desktop/operators$
```


Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>



The screenshot shows a VS Code editor window with a file named `main.nf` open. The code in the editor is a Nextflow DSL script:

```
main.nf
1  ch = channel
2    .fromPath('data/yeast/reads/*.fq.gz')
3    .view()
```

Below the editor, the `TERMINAL` tab is active, showing the output of the command `nextflow run main.nf`. The output indicates that Nextflow version 24.10.4 is being used and that the workflow `main.nf` (revision 74ef96805e) is being launched. The output lists 20 files from the `data/yeast/reads/` directory, including `ref2_1.fq.gz`, `etoh60_3_1.fq.gz`, `ref3_1.fq.gz`, `etoh60_3_2.fq.gz`, `temp33_3_1.fq.gz`, `etoh60_1_1.fq.gz`, `temp33_1_2.fq.gz`, `etoh60_2_2.fq.gz`, `temp33_2_2.fq.gz`, `temp33_2_1.fq.gz`, `etoh60_2_1.fq.gz`, `ref1_2.fq.gz`, `ref3_2.fq.gz`, `ref1_1.fq.gz`, `etoh60_1_2.fq.gz`, `temp33_1_1.fq.gz`, `ref2_2.fq.gz`, and `temp33_3_2.fq.gz`.

```
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [deadly_dubinsky] DSL2 - revision: 74ef96805e

/home/antoine/Desktop/operators/data/yeast/reads/ref2_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/etoh60_3_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/ref3_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/etoh60_3_2.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/temp33_3_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/etoh60_1_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/temp33_1_2.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/etoh60_2_2.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/temp33_2_2.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/temp33_2_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/etoh60_2_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/ref1_2.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/ref3_2.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/ref1_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/etoh60_1_2.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/temp33_1_1.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/ref2_2.fq.gz
/home/antoine/Desktop/operators/data/yeast/reads/temp33_3_2.fq.gz
antoine@antoine-Lemur:~/Desktop/operators$
```

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

String manipulation

The screenshot shows a Nextflow workflow editor with a file named `main.nf`. The workflow code is as follows:

```
1 ch = channel
2   .fromPath('data/yeast/reads/*.fq.gz')
3   .map { file -> [ "Hello!! 😊", file ] }
4   .view()
```

A red dashed box highlights the `map` operator on line 3. A red arrow points to the `file` variable in the lambda function, indicating string manipulation.

The terminal output shows the execution of the workflow:

```
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [ridiculous_edison] DSL2 - revision: 67c7f5567e

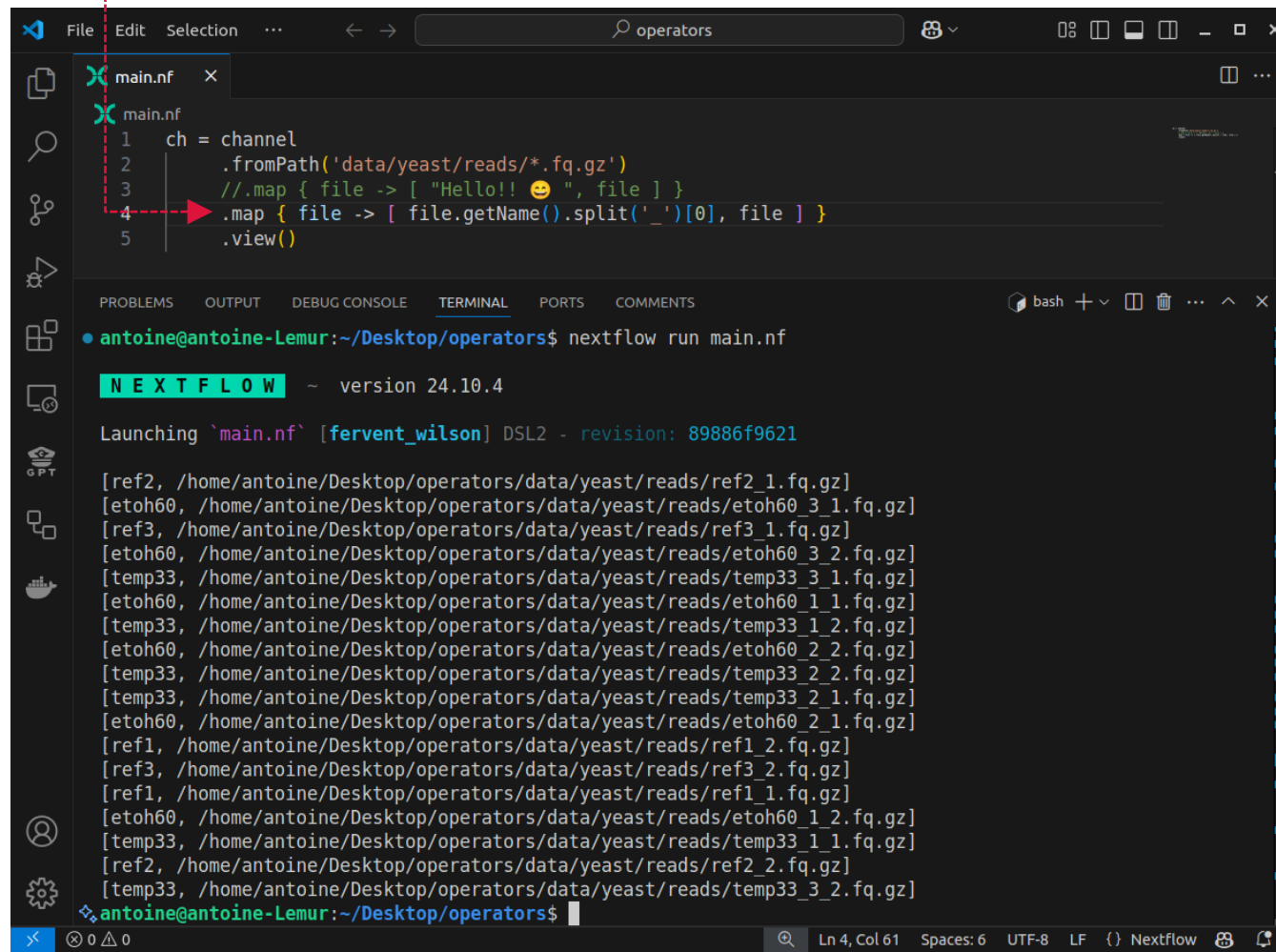
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/ref2_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/etoh60_3_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/ref3_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/etoh60_3_2.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/temp33_3_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/etoh60_1_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/temp33_1_2.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/etoh60_2_2.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/temp33_2_2.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/temp33_2_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/etoh60_2_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/ref1_2.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/ref3_2.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/ref1_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/etoh60_1_2.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/temp33_1_1.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/ref2_2.fq.gz]
[Hello!! 😊 , /home/antoine/Desktop/operators/data/yeast/reads/temp33_3_2.fq.gz]
antoine@antoine-Lemur:~/Desktop/operators$
```

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

String manipulation



The screenshot shows a VS Code editor window with a file named `main.nf`. The script content is as follows:

```
1 ch = channel
2   .fromPath('data/yeast/reads/*.fq.gz')
3   //.map { file -> [ "Hello!! 😊", file ] }
4   .map { file -> [ file.getName().split('_')[0], file ] }
5   .view()
```

A red dashed box highlights the `main.nf` file in the Explorer and the corresponding line in the editor. A red arrow points from the Explorer to the editor.

The terminal window below the editor shows the execution of the script:

```
• antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [fervent_wilson] DSL2 - revision: 89886f9621

[ref2, /home/antoine/Desktop/operators/data/yeast/reads/ref2_1.fq.gz]
[etoh60, /home/antoine/Desktop/operators/data/yeast/reads/etoh60_3_1.fq.gz]
[ref3, /home/antoine/Desktop/operators/data/yeast/reads/ref3_1.fq.gz]
[etoh60, /home/antoine/Desktop/operators/data/yeast/reads/etoh60_3_2.fq.gz]
[temp33, /home/antoine/Desktop/operators/data/yeast/reads/temp33_3_1.fq.gz]
[etoh60, /home/antoine/Desktop/operators/data/yeast/reads/etoh60_1_1.fq.gz]
[temp33, /home/antoine/Desktop/operators/data/yeast/reads/temp33_1_2.fq.gz]
[etoh60, /home/antoine/Desktop/operators/data/yeast/reads/etoh60_2_2.fq.gz]
[temp33, /home/antoine/Desktop/operators/data/yeast/reads/temp33_2_2.fq.gz]
[temp33, /home/antoine/Desktop/operators/data/yeast/reads/temp33_2_1.fq.gz]
[etoh60, /home/antoine/Desktop/operators/data/yeast/reads/etoh60_2_1.fq.gz]
[ref1, /home/antoine/Desktop/operators/data/yeast/reads/ref1_2.fq.gz]
[ref3, /home/antoine/Desktop/operators/data/yeast/reads/ref3_2.fq.gz]
[ref1, /home/antoine/Desktop/operators/data/yeast/reads/ref1_1.fq.gz]
[etoh60, /home/antoine/Desktop/operators/data/yeast/reads/etoh60_1_2.fq.gz]
[temp33, /home/antoine/Desktop/operators/data/yeast/reads/temp33_1_1.fq.gz]
[ref2, /home/antoine/Desktop/operators/data/yeast/reads/ref2_2.fq.gz]
[temp33, /home/antoine/Desktop/operators/data/yeast/reads/temp33_3_2.fq.gz]
```

The status bar at the bottom indicates the current position is Ln 4, Col 61, with 6 spaces, UTF-8 encoding, LF line endings, and the Nextflow DSL2 language.

Operators

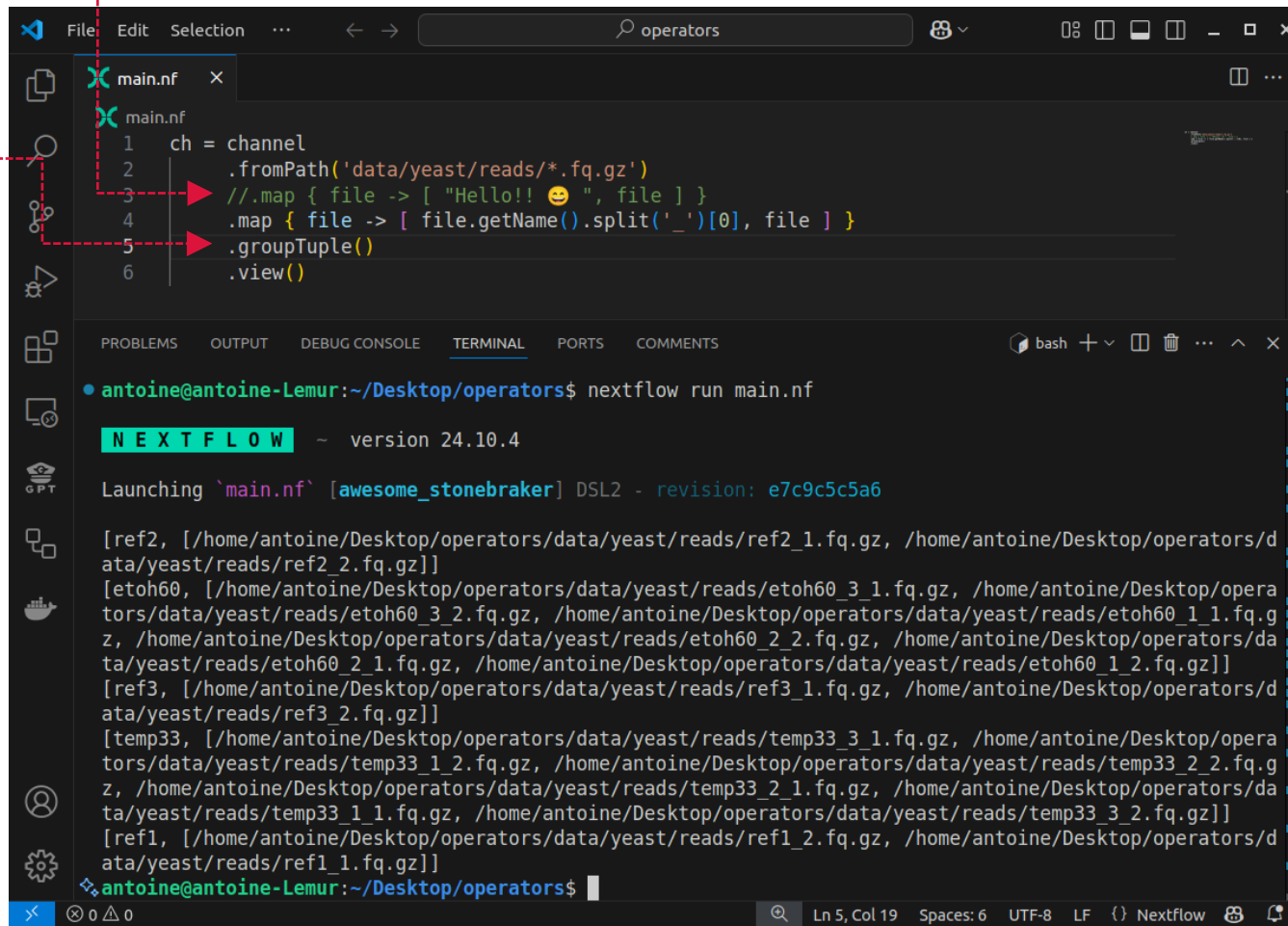
Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

String manipulation

+

Group stuff together!
Such as read pairs



The screenshot shows a VS Code editor with a file named `main.nf` open. The script content is as follows:

```
1 ch = channel
2   .fromPath('data/yeast/reads/*.fq.gz')
3   .map { file -> [ "Hello!! 😊 ", file ] }
4   .map { file -> [ file.getName().split('_')[0], file ] }
5   .groupTuple()
6   .view()
```

Red dashed lines and arrows point from the text annotations to the script. One arrow points from 'String manipulation' to the `split('_')` operation on line 4. Another arrow points from 'Group stuff together! Such as read pairs' to the `groupTuple()` operation on line 5.

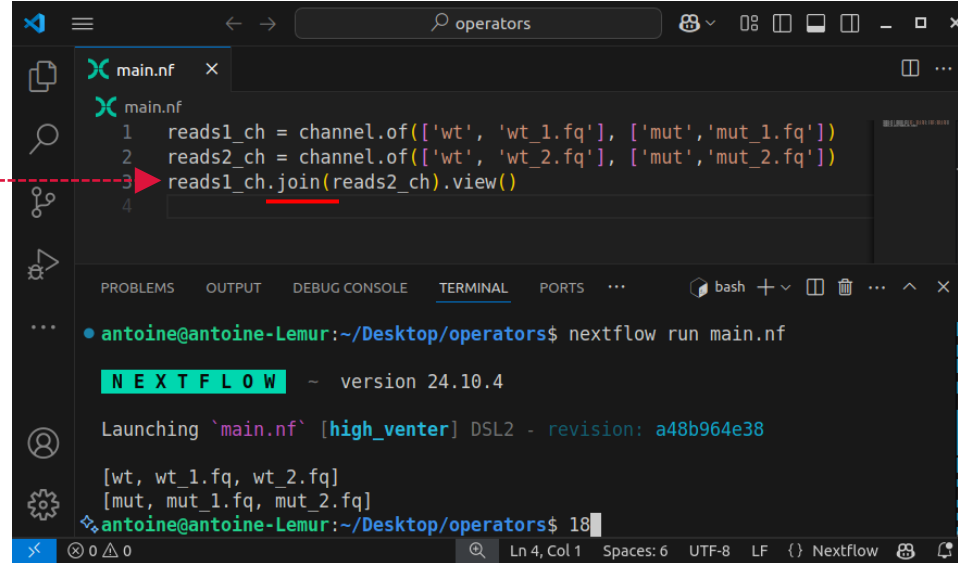
Below the editor, the terminal shows the command `nextflow run main.nf` being executed. The output displays the Nextflow version (24.10.4) and the DSL2 revision (e7c9c5c5a6). It then lists the paths for various read pairs, such as `[ref2, [/home/antoine/Desktop/operators/data/yeast/reads/ref2_1.fq.gz, /home/antoine/Desktop/operators/data/yeast/reads/ref2_2.fq.gz]]`.

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Join lists !
(also mix)



The screenshot shows a VS Code editor with a file named `main.nf` open. The script contains the following code:

```
1 reads1_ch = channel.of(['wt', 'wt_1.fq'], ['mut', 'mut_1.fq'])
2 reads2_ch = channel.of(['wt', 'wt_2.fq'], ['mut', 'mut_2.fq'])
3 reads1_ch.join(reads2_ch).view()
4
```

A red dashed line points from the text "Join lists ! (also mix)" to the `join` method in line 3. The terminal below shows the execution of the script:

```
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [high_venter] DSL2 - revision: a48b964e38

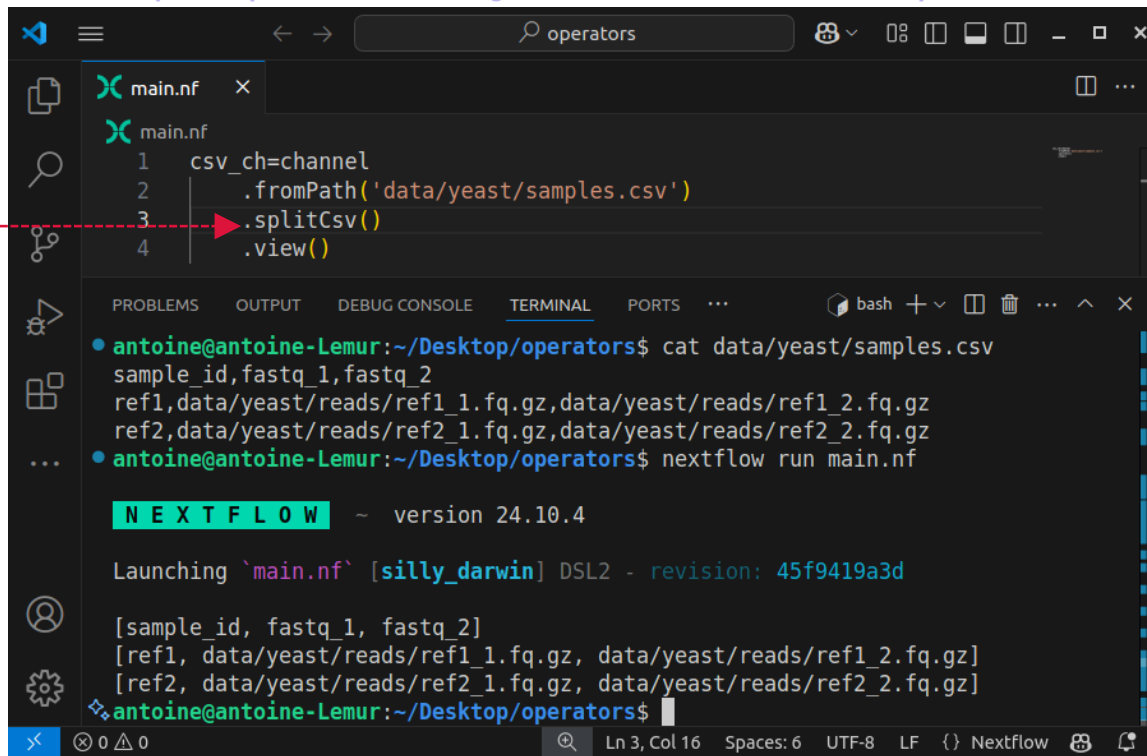
[wt, wt_1.fq, wt_2.fq]
[mut, mut_1.fq, mut_2.fq]
antoine@antoine-Lemur:~/Desktop/operators$
```

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Extract file content



The screenshot shows a VS Code editor with a file named `main.nf` open. The script contains the following code:

```
1 csv_ch=channel
2   .fromPath('data/yeast/samples.csv')
3   .splitCsv()
4   .view()
```

A red dashed line points from the text "Extract file content" to the `.splitCsv()` operator on line 3. The terminal at the bottom shows the execution of the script:

```
antoine@antoine-Lemur:~/Desktop/operators$ cat data/yeast/samples.csv
sample_id,fastq_1,fastq_2
ref1,data/yeast/reads/ref1_1.fq.gz,data/yeast/reads/ref1_2.fq.gz
ref2,data/yeast/reads/ref2_1.fq.gz,data/yeast/reads/ref2_2.fq.gz
...
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [silly_darwin] DSL2 - revision: 45f9419a3d

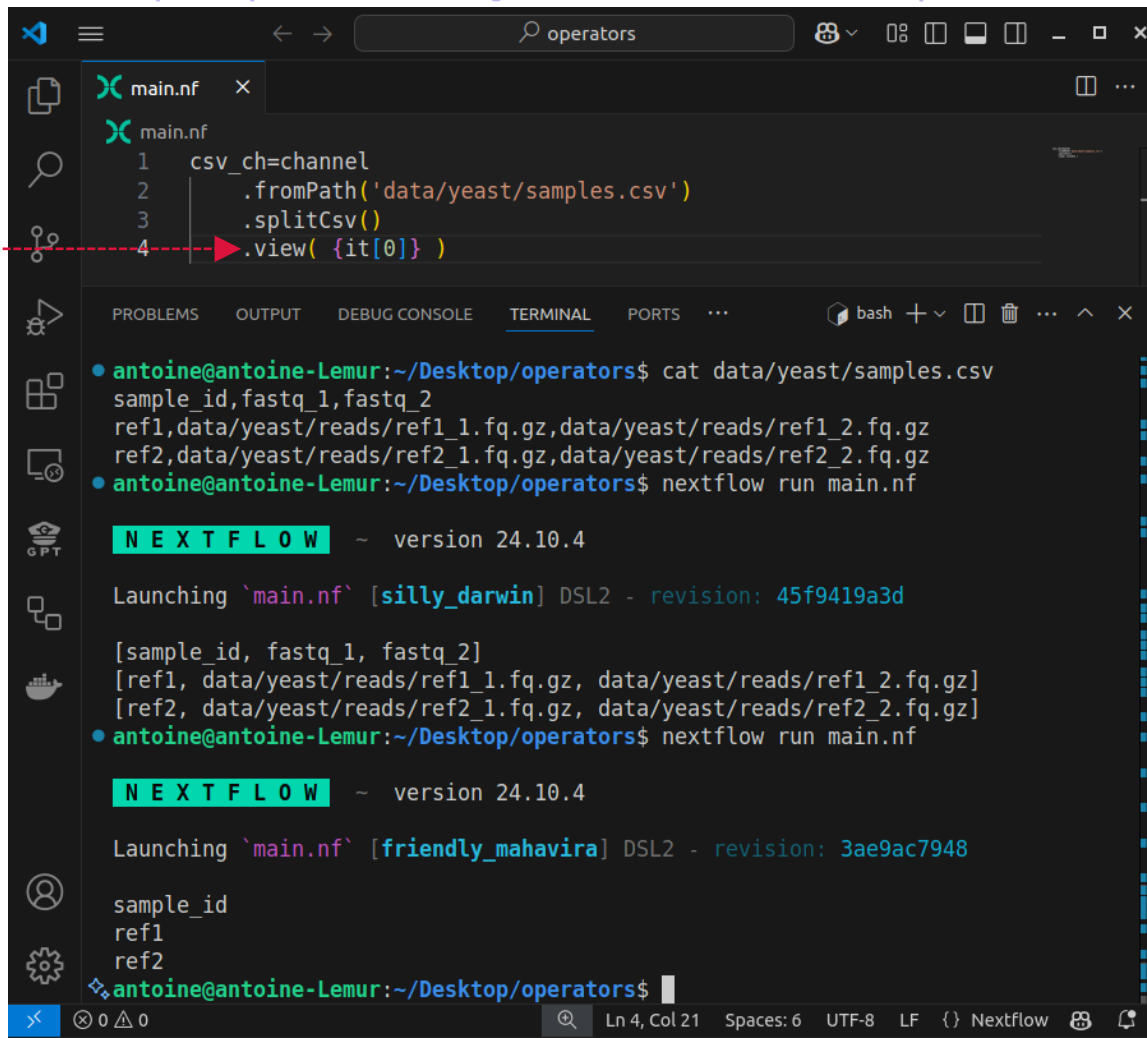
[sample_id, fastq_1, fastq_2]
[ref1, data/yeast/reads/ref1_1.fq.gz, data/yeast/reads/ref1_2.fq.gz]
[ref2, data/yeast/reads/ref2_1.fq.gz, data/yeast/reads/ref2_2.fq.gz]
antoine@antoine-Lemur:~/Desktop/operators$
```

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

Extract file content
+ further manipulation



The screenshot shows a Nextflow workflow editor with a file named `main.nf`. The workflow code is as follows:

```
1 csv_ch=channel
2   .fromPath('data/yeast/samples.csv')
3   .splitCsv()
4   .view( {it[0]} )
```

A red dashed line points from the text "Extract file content + further manipulation" to the `.view({it[0]})` line in the workflow code.

The terminal output shows the execution of the workflow:

```
antoine@antoine-Lemur:~/Desktop/operators$ cat data/yeast/samples.csv
sample_id,fastq_1,fastq_2
ref1,data/yeast/reads/ref1_1.fq.gz,data/yeast/reads/ref1_2.fq.gz
ref2,data/yeast/reads/ref2_1.fq.gz,data/yeast/reads/ref2_2.fq.gz
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [silly_darwin] DSL2 - revision: 45f9419a3d

[sample_id, fastq_1, fastq_2]
[ref1, data/yeast/reads/ref1_1.fq.gz, data/yeast/reads/ref1_2.fq.gz]
[ref2, data/yeast/reads/ref2_1.fq.gz, data/yeast/reads/ref2_2.fq.gz]
antoine@antoine-Lemur:~/Desktop/operators$ nextflow run main.nf

NEXTFLOW ~ version 24.10.4

Launching `main.nf` [friendly_mahavira] DSL2 - revision: 3ae9ac7948

sample_id
ref1
ref2
antoine@antoine-Lemur:~/Desktop/operators$
```

Operators

Overview on some operators

→ <https://carpentries-incubator.github.io/workflows-nextflow/07-operators.html>

More useful examples

→ https://training.nextflow.io/latest/basic_training/operators/

Thank you for your attention!

Want to go further?
Ask us to organize Nextflow events, hackathons, ...



zurich-data-scientists.ch



nextflow.io/our_ambassadors.html

More info



<https://nf-co.re>



github.com/nf-core



nfcore.slack.com



youtube.com/nf-core



@nf_core@mstdn.science



[@nf_core](https://twitter.com/nf_core)