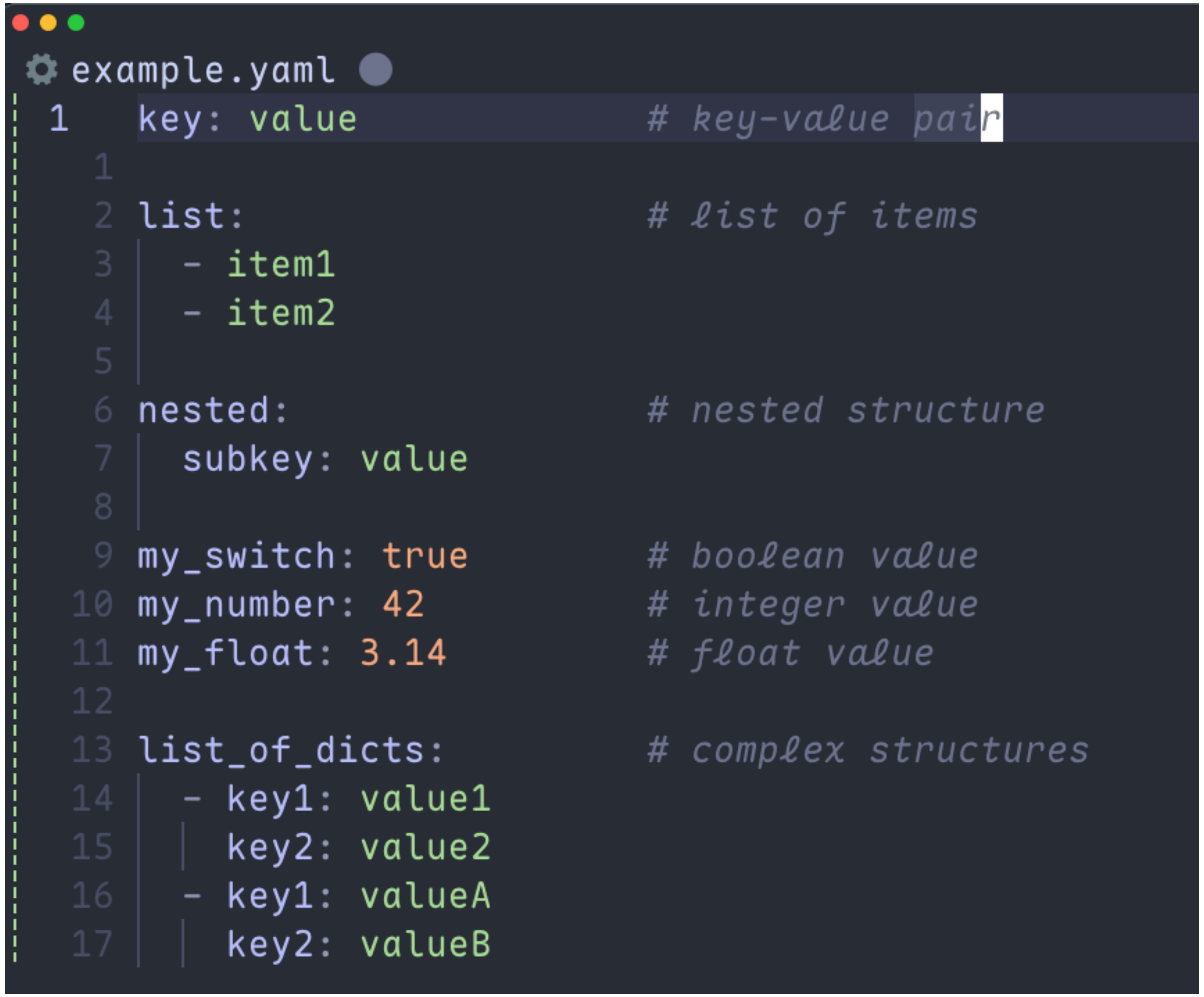
PyMOR Syntax

How to put together your YAML files

Paul Gierz

A very, very quick introduction to YAML

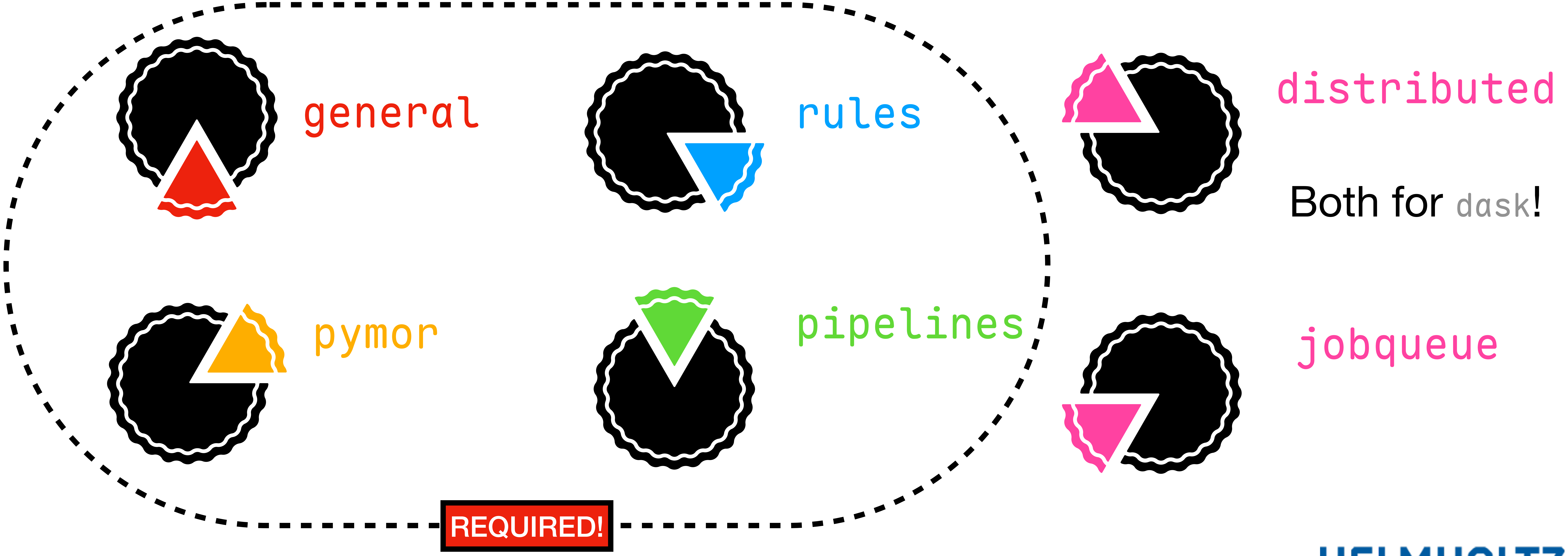
- YAML = "YAML Ain’t Markup Language" (human-friendly config format)
- Used for: configuration files, data exchange (e.g. Docker, GitHub Actions)
- Rules:
  - **Indentation matters** (use spaces, not tabs)
  - No **commas**, **brackets**, or **quotes** needed (usually)
  - Supports **strings**, **numbers**, **booleans**, **lists**, **dictionaries**



```yaml
key: value             # key-value pair

list:                  # list of items
  - item1
  - item2

nested:                # nested structure
  subkey: value

my_switch: true        # boolean value
my_number: 42          # integer value
my_float: 3.14         # float value

list_of_dicts:         # complex structures
  - key1: value1
    key2: value2
  - key1: valueA
    key2: valueB
```

Sections of a PyMOR YAML File

- Our YAML files are divided into several sections:



`general`

`rules`

`pymor`

`pipelines`

REQUIRED!

`distributed`

Both for `dask`!

`jobqueue`

general

Sections of a PyMOR YAML File

General Section

- Contains information about the PyMOR configuration file (name, description, maintainer, email, etc). **Required keys** here are:

  1. *`cmor_version`*: Which CMOR version this file is prepared for (CMIP6 or CMIP7)
  2. *`CMIP_Tables_Dir`*: location of JSON Tables
  3. *`CV_Dir`*: location of Controlled Vocabularies

Sections of a PyMOR YAML File

Pymor Section

- Contains information about how PyMOR should operate. The following keys are of interest:

  1. `warn_on_no_rule` Should PyMOR warn you about CMOR variables found in `CMIP_Tables_Dir` that don't have a rule
  2. `dask_cluster`: What type of Dask cluster to spin up (local or on SLURM)
  3. `dask_cluster_scaling_mode`: If the cluster supports scaling, do you want to use adaptive or fixed scaling?
  4. `dask_cluster_scaling_fixed_jobs`: If you run in fixed mode, how many SLURM jobs containing Dask workers do you want to create?

- Can also be controlled via environment variables! Run config takes precedence over env variable

Sections of a PyMOR YAML File

Rules Section

`rules`

- Contains a list of dictionaries which are converted into `pymor.core.rule.Rule` objects, mapping how to create each CMOR requested variable!
- Must have
  - `cmor_variable`
  - `inputs`
  - `output_directory`
  - `model_component`
  - `grid_label`
  - `experiment_id`
  - `model_variable`
  - `pipelines`
  - `variant_label`
  - `source_id`
- Validated with useful error messages showing where the object is misconfigured if you have any problems!

Sections of a PyMOR YAML File

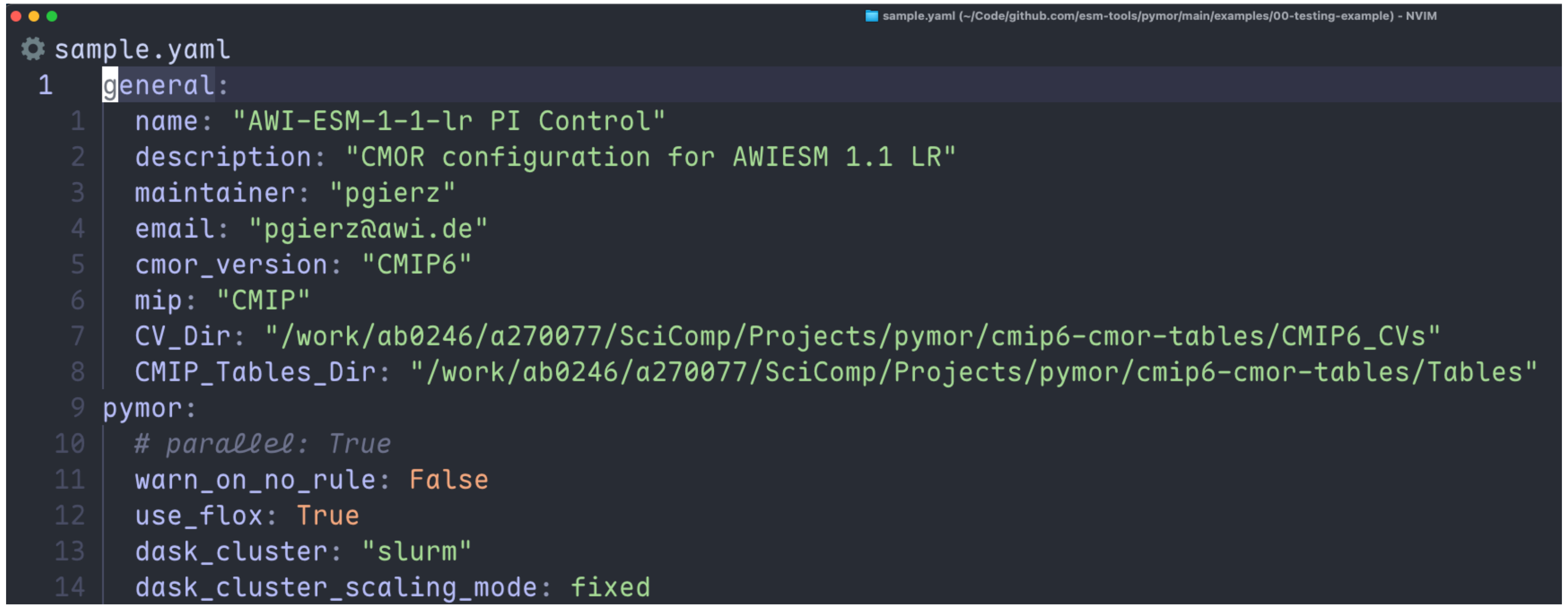
Pipelines Section

- Contains a list of dictionaries which are converted into `pymor.core.pipeline.Pipeline` objects, what steps to take when processing a particular data set through PyMOR.

- Must have
  - `name`
  - `steps: list of Python qualified names`
  - `uses: an importable Pipeline object`

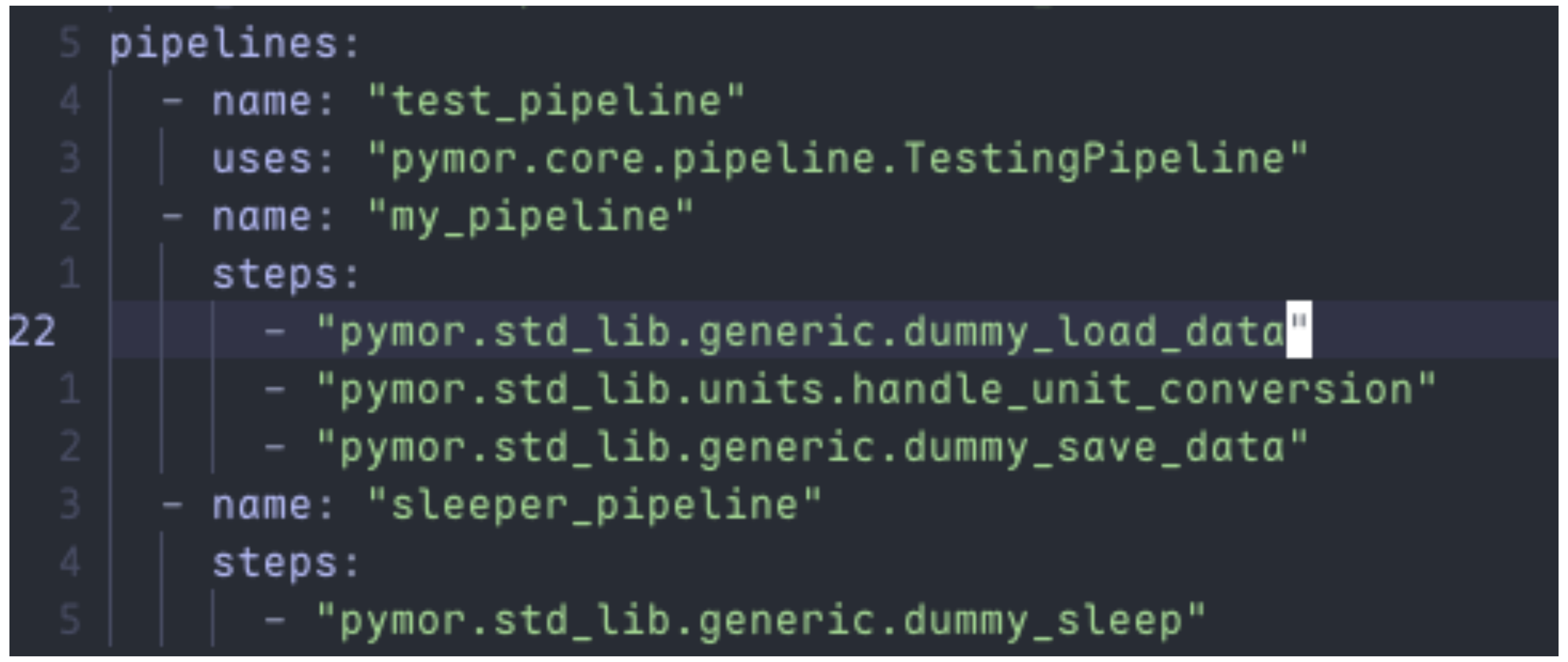
- Validated with useful error messages showing where the object is misconfigured if you have any problems!

Some Examples

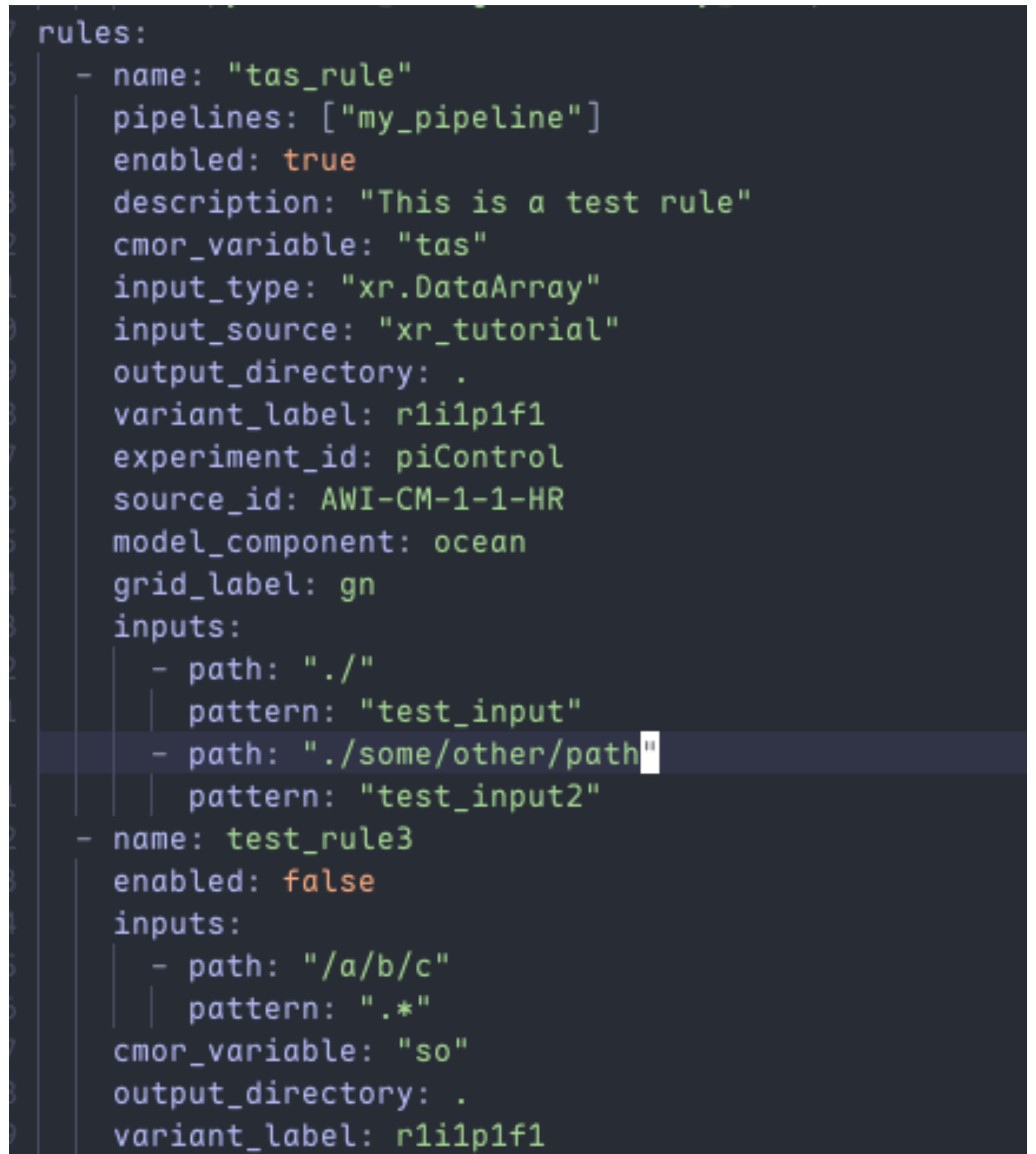


sample.yaml (~/Code/github.com/esm-tools/pymor/main/examples/00-testing-example) - NVIM

```yaml
sample.yaml
general:
  name: "AWI-ESM-1-1-lr PI Control"
  description: "CMOR configuration for AWIESM 1.1 LR"
  maintainer: "pgierz"
  email: "pgierz@awi.de"
  cmor_version: "CMIP6"
  mip: "CMIP"
  CV_Dir: "/work/ab0246/a270077/SciComp/Projects/pymor/cmip6-cmor-tables/CMIP6_CVs"
  CMIP_Tables_Dir: "/work/ab0246/a270077/SciComp/Projects/pymor/cmip6-cmor-tables/Tables"
pymor:
  # parallel: True
  warn_on_no_rule: False
  use_flox: True
  dask_cluster: "slurm"
  dask_cluster_scaling_mode: fixed
```



```yaml
pipelines:
  - name: "test_pipeline"
    uses: "pymor.core.pipeline.TestingPipeline"
  - name: "my_pipeline"
    steps:
      - "pymor.std_lib.generic.dummy_load_data"
      - "pymor.std_lib.units.handle_unit_conversion"
      - "pymor.std_lib.generic.dummy_save_data"
  - name: "sleeper_pipeline"
    steps:
      - "pymor.std_lib.generic.dummy_sleep"
```



```yaml
rules:
  - name: "tas_rule"
    pipelines: ["my_pipeline"]
    enabled: true
    description: "This is a test rule"
    cmor_variable: "tas"
    input_type: "xr.DataArray"
    input_source: "xr_tutorial"
    output_directory: .
    variant_label: r1i1p1f1
    experiment_id: piControl
    source_id: AWI-CM-1-1-HR
    model_component: ocean
    grid_label: gn
    inputs:
      - path: "./"
        pattern: "test_input"
      - path: "./some/other/path"
        pattern: "test_input2"
  - name: test_rule3
    enabled: false
    inputs:
      - path: "/a/b/c"
        pattern: ".*"
    cmor_variable: "so"
    output_directory: .
    variant_label: r1i1p1f1
```

yq can be helpful!

```yaml
test_config_cmip6.yaml
pymor:
  version: "unreleased"
  use_xarray_backend: True
  warn_on_no_rule: False
  minimum_jobs: 8
  maximum_jobs: 10
general:
  name: "test"
```

Full files can get very long!