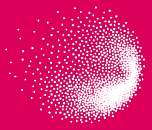


PSI Center for Scientific Computing, Theory and Data

Writing workflows

An outsider’s perspective

Edward Linscott

LMS Seminar, 12 March 2025

Outline

Outline

What I’ve learned after 5 years of trying to automate Koopmans functionals https://knowyourmeme.com/memes/abe-simpson-talking-to-kids

Well not quite

Outline

but the intervening years have not been kind – see permit photos)

History of koopmans

- I want to run Koopmans functional calculations and I know how to write python and use ASE (and I was advised not to use AiiDA…)
- atoms
- calculators
- where necessary, use outputs of previous calculation into subsequent calculation (e.g. link a file, set a parameter etc.)
- start writing multiple scripts
- wannierisation
- dscf
- dfpt

History of koopmans

- natural emergence of the idea of a workflow and subworkflows that I want to be able to reuse

… koopmans

Interfacing with AiiDA

What I needed to do

- isolate into steps
- functional programming

COMMON WORKFLOW LANGUAGE

COMMON WORKFLOW LANGUAGE

- “an open standard for describing how to run command line tools and connect them to create workflows”

COMMON WORKFLOW LANGUAGE

- “an open standard for describing how to run command line tools and connect them to create workflows”
- introduced in 2014; version 1.2 released in 2020

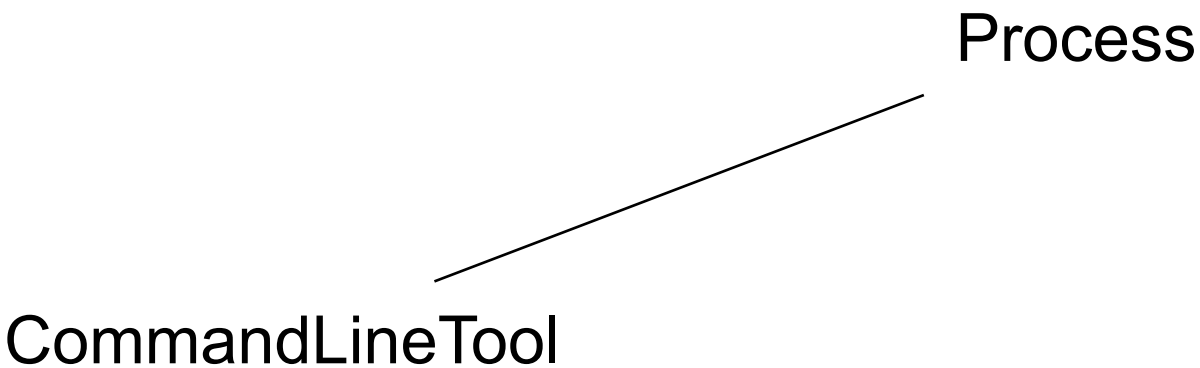
COMMON WORKFLOW LANGUAGE

- “an open standard for describing how to run command line tools and connect them to create workflows”
- introduced in 2014; version 1.2 released in 2020
- mostly used by bioinformatics community

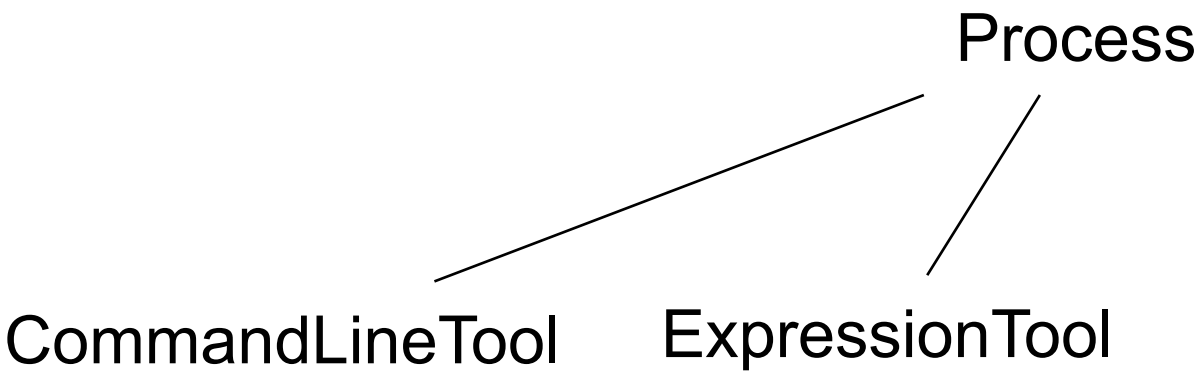
Basic Concepts

Process

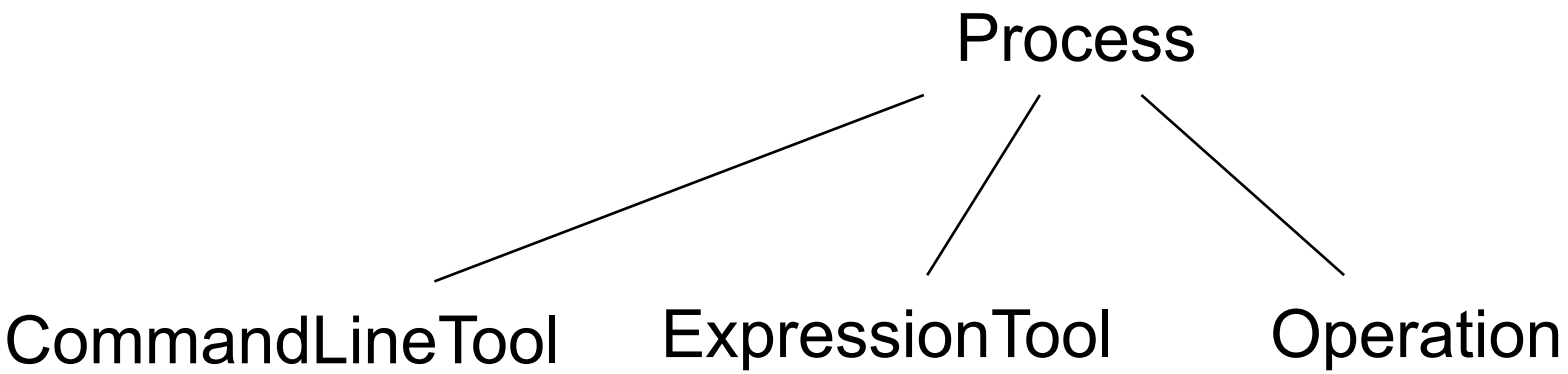
Basic Concepts



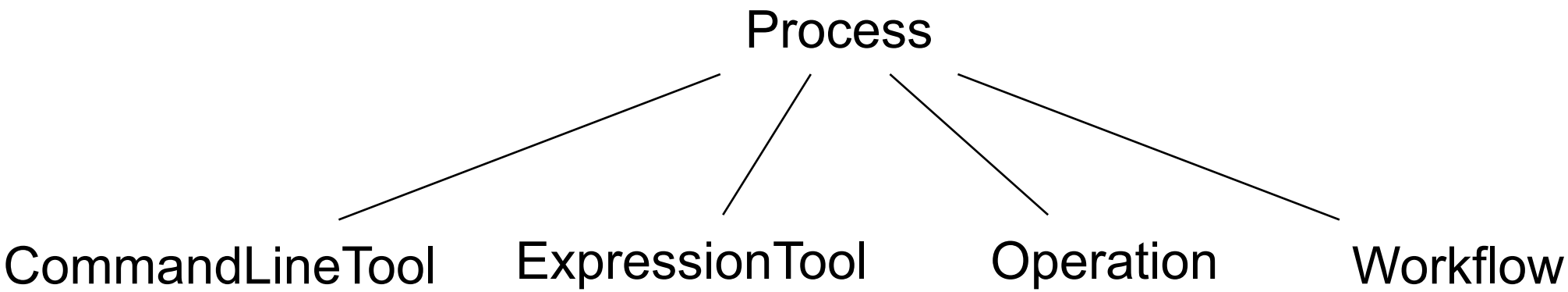
Basic Concepts



Basic Concepts



Basic Concepts



CommandLineTool

PSI

echo.cwl

```yaml
cwlVersion: v1.2
class: CommandLineTool
baseCommand: echo

inputs:
  message:
    type: string
    default: "Hello World"
    inputBinding:
      position: 1

outputs: []
```

ExpressionTool

PSI

uppercase.cwl

```yaml
cwlVersion: v1.2
class: ExpressionTool
requirements:
  InlineJavascriptRequirement: {}

inputs:
  message: string
outputs:
  uppercase_message: string

expression: |
  ${ return {"uppercase_message": inputs.message.toUpperCase()}; }
```

Workflow

PSI

echo_uppercase.cwl

```yaml
cwlVersion: v1.2
class: Workflow

requirements:
  InlineJavascriptRequirement: {}

inputs:
  message: string

outputs:
  out:
    type: string
    outputSource: uppercase/uppercase_message
```

Workflow

```yaml
steps:
  echo:
    run: echo.cwl
    in:
      message: message
    out: [out]
  uppercase:
    run: uppercase.cwl
    in:
      message:
        source: echo/out
    out: [uppercase_message]
```

Pros and Cons

- pros:

Pros and Cons

- pros:
  - clear and explicit

Pros and Cons

Pros and Cons

- pros:
  - clear and explicit
  - composable and customisable

Pros and Cons

- pros:
  - clear and explicit
  - composable and customisable
- cons:

Pros and Cons

- pros:
  - clear and explicit
  - composable and customisable
- cons:
  - verbose

Pros and Cons

- pros:
  - clear and explicit
  - composable and customisable
- cons:
  - verbose
  - complicated workflows lead to very complicated CWL (e.g. `while`)

Pros and Cons

- pros:
  - clear and explicit
  - composable and customisable
- cons:
  - verbose
  - complicated workflows lead to very complicated CWL (e.g. `while`)
  - `ExpressionTool` restricted to Javascript

Pros and Cons

- pros:
  - clear and explicit
  - composable and customisable
- cons:
  - verbose
  - complicated workflows lead to very complicated CWL (e.g. `while`)
  - `ExpressionTool` restricted to Javascript
  - need to define custom types (e.g. OPTIMADE, PREMISE)

Pros and Cons

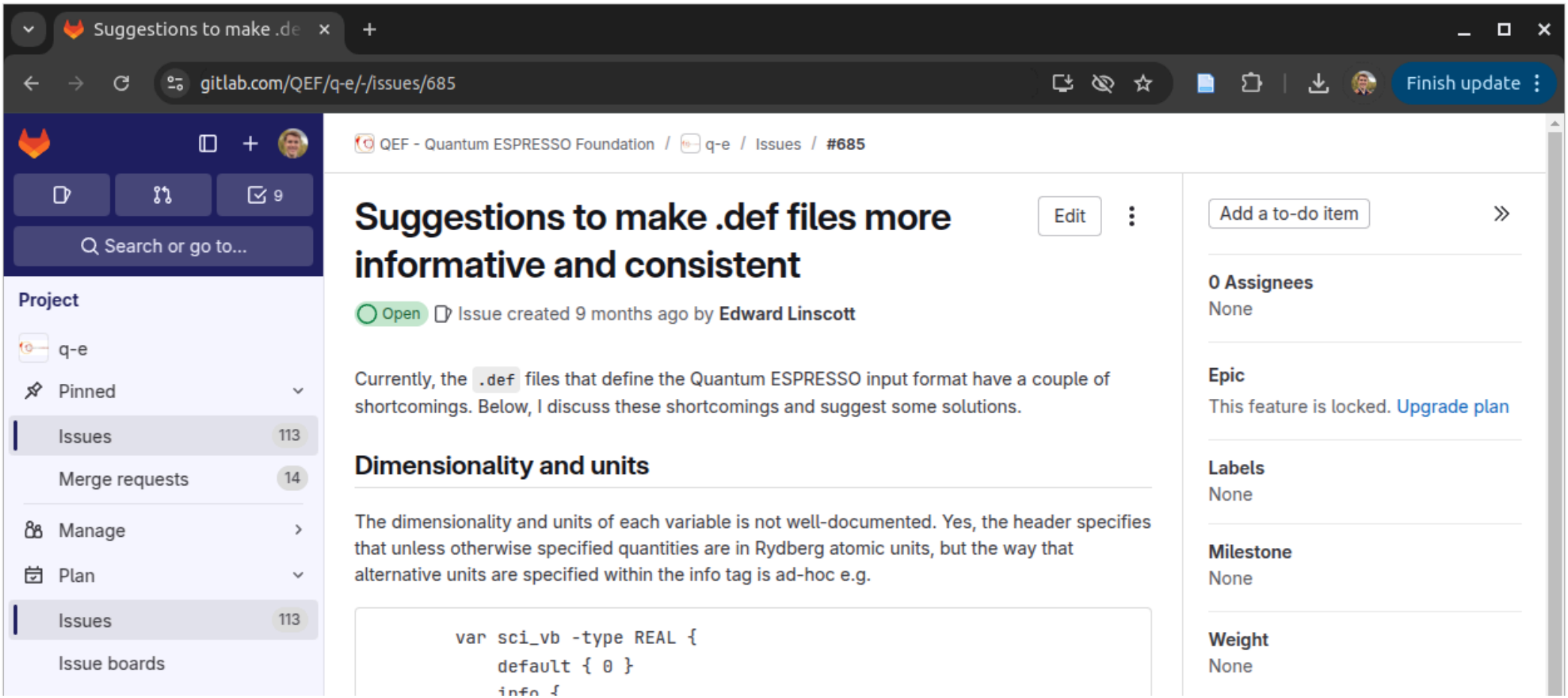
- pros:
  - clear and explicit
  - composable and customisable
- cons:
  - verbose
  - complicated workflows lead to very complicated CWL (e.g. `while`)
  - `ExpressionTool` restricted to Javascript
  - need to define custom types (e.g. OPTIMADE, PREMISE)
  - custom types do not permit defaults

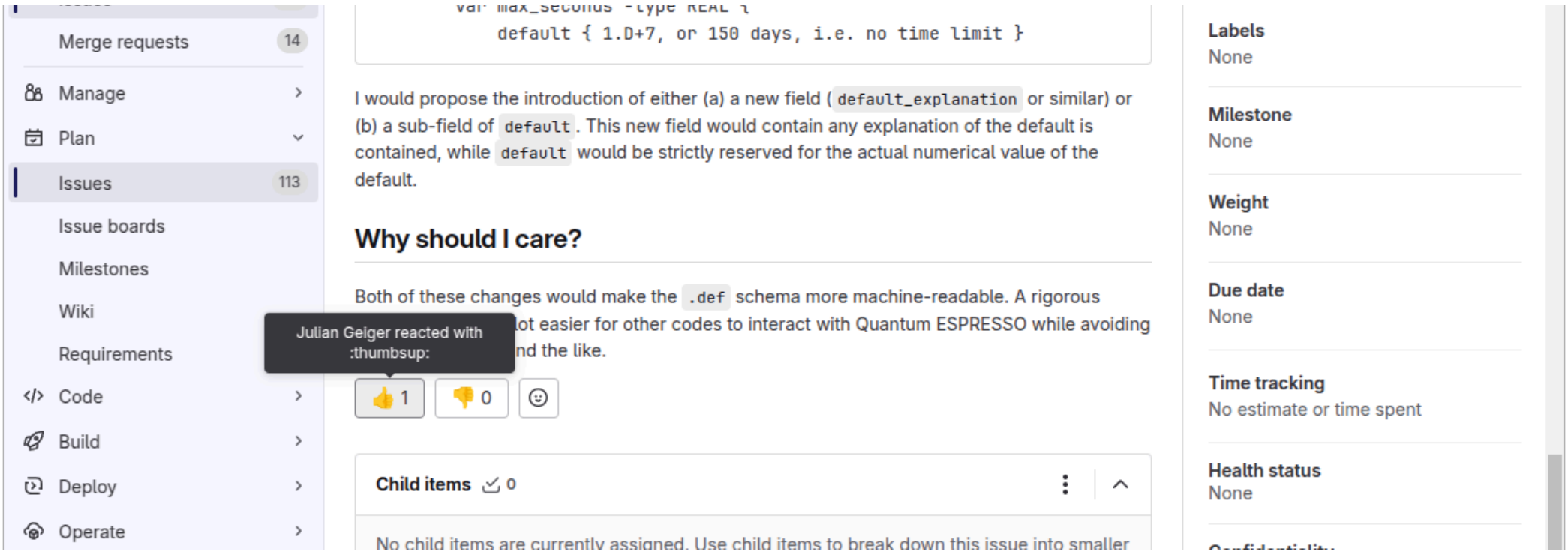
Pros and Cons

- pros:
  - clear and explicit
  - composable and customisable
- cons:
  - verbose
  - complicated workflows lead to very complicated CWL (e.g. `while`)
  - `ExpressionTool` restricted to Javascript
  - need to define custom types (e.g. OPTIMADE, PREMISE)
  - custom types do not permit defaults
  - rigorous schemas require willingness from the community

Pros and Cons

- pros:
  - clear and explicit
  - composable and customisable
- cons:
  - verbose
  - complicated workflows lead to very complicated CWL (e.g. `while`)
  - `ExpressionTool` restricted to Javascript
  - need to define custom types (e.g. OPTIMADE, PREMISE)
  - custom types do not permit defaults
  - rigorous schemas require willingness from the community


Suggestions to make .de
gitlab.com/QEF/q-e/-/issues/685
Finish update
Search or go to...
Project
q-e
Pinned
Issues 113
Merge requests 14
Manage
Plan
Issues 113
Issue boards
QEF - Quantum ESPRESSO Foundation / q-e / Issues / #685
Suggestions to make .def files more informative and consistent
Edit
Open
Issue created 9 months ago by Edward Linscott
Currently, the .def files that define the Quantum ESPRESSO input format have a couple of shortcomings. Below, I discuss these shortcomings and suggest some solutions.
Dimensionality and units
The dimensionality and units of each variable is not well-documented. Yes, the header specifies that unless otherwise specified quantities are in Rydberg atomic units, but the way that alternative units are specified within the info tag is ad-hoc e.g.
var sci_vb -type REAL {
default { 0 }
Add a to-do item
0 Assignees
None
Epic
This feature is locked. Upgrade plan
Labels
None
Milestone
None
Weight
None


Merge requests 14
Manage
Plan
Issues 113
Issue boards
Milestones
Wiki
Requirements
Code
Build
Deploy
Operate
default { 1.D+7, or 150 days, i.e. no time limit }
I would propose the introduction of either (a) a new field (default_explanation or similar) or (b) a sub-field of default . This new field would contain any explanation of the default is contained, while default would be strictly reserved for the actual numerical value of the default.
Why should I care?
Both of these changes would make the .def schema more machine-readable. A rigorous
ot easier for other codes to interact with Quantum ESPRESSO while avoiding
nd the like.
Julian Geiger reacted with :thumbsup:
1
0
Child items 0
No child items are currently assigned. Use child items to break down this issue into smaller
Labels
None
Milestone
None
Weight
None
Due date
None
Time tracking
No estimate or time spent
Health status
None

So where does that leave koopmans?

Rewriting `koopmans`

`koopmans` is slowly being refactored into “CWL-inspired” python

```python
class Bin2XMLInput(IOModel):
    binary: File

    class Config:
        arbitrary_types_allowed = True


class Bin2XMLOutput(IOModel):
    xml: File

    class Config:
        arbitrary_types_allowed = True


class Bin2XMLProcess(CommandLineTool):

    input_model = Bin2XMLInput
    output_model = Bin2XMLOutput

    def _pre_run(self):
        super()._pre_run()
```

Rewriting koopmans

```
        if not self.inputs.binary.exists():
            raise FileNotFoundError(f'`{self.inputs.binary}` does not exist')

        # Link the input binary file to the directory of this process as input.dat
        dst = File(self, Path("input.dat"))
        dst.symlink_to(self.inputs.binary)

    @property
    def command(self):
        return Command(executable='bin2xml.x', suffix=f'input.dat output.xml')

    def _set_outputs(self):
        xml_filepointer = File(self, Path("output.xml"))
        self.outputs = self.output_model(xml=xml_filepointer)
```

Aside: Common Workflows

Common Workflows

Compare with Common Workflows (all implemented in AiiDA https://doi.org/10.1038/s41524-021-00594-6, but can change code – maybe talk with Marnik about this and the advantages/limitations. Are people writing any common workflows or is everything still code-dependent? No – for anything complicated you need code-specific logic)

In an ideal world:

- workflows with very prescribed inputs and outputs
- workflows that are engine-agnostic (we should not have to rewrite how to calculate binding energy curves, defect energies etc again and again – nor should this knowledge exclusive to AiiDA)

Common Workflows

- should make concatenating workflows straightforward
- workflow manages such as AiiDA can read and run .cwl files

The alternative: siloed communities where we only write AiiDA – that is valid, but we need to simplify, we need to educate, we need to commit (and not push people away from it)

Where we call down

- everyone wants bespoke

Test[1]

[1]This is a footnote

Introduction

References

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam

References

et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Here is a focus slide presenting a key idea

This is a matrix slide

You can use it to present information side-by-side

with an arbitrary number of rows and columns

Test

More text appears under the same subsection title as earlier

But a new subsection starts a new page.

Now, let’s cite a nice paper.[1]

[1] E. B. Linscott *et al. J. Chem. Theory Comput.* **19,** 7097–7111 (2023)

Test

E. B. Linscott *et al.* koopmans: an open-source package for accurately and efficiently predicting spectral properties with Koopmans Functionals. *J. Chem. Theory Comput.* **19,** 7097–7111 (2023).