${} – variable calling

Syntax

The value of any yaml variable can be retrieve in any file involve in the given operation (for AWI-ESM-2.1, those files are <your_runscript>.yaml, fesom-2.1.yaml, oasis3mct.yaml, echam.yaml, <machine>.yaml, <slurm/pbs>.yaml):

- In the same section

<your_runscript>.yaml

```
fesom:
    foo1: bar
    foo2: ${foo1}
```

- Independent of the order

<your_runscript>.yaml

```
fesom:
    foo2: ${foo1}
    …
    foo1: bar
```

${} – variable calling

Syntax

The value of any yaml variable can be retrieve in any file involve in the given operation (for AWI-ESM-2.1, those files are <your_runscript>.yaml, fesom-2.1.yaml, oasis3mct.yaml, echam.yaml, <machine>.yaml, <slurm/pbs>.yaml):

- In across different files

fesom-2.1.yaml

```
fesom:
    foo1: bar
```

awiesm.yaml

```
fesom:
    foo2: ${foo1}
```

- Across different sections using the syntax ${<section>.<variable>}

<your_runscript>.yaml

```
general:
    foo1: bar
fesom:
    foo2: ${general.foo1}
```

${} – variable calling

Syntax

The value of any yaml variable can be retrieve in any file involve in the given operation (for AWI-ESM-2.1, those files are <your_runscript>.yaml, fesom-2.1.yaml, oasis3mct.yaml, echam.yaml, <machine>.yaml, <slurm/pbs>.yaml):

- Independently of the type (as long as it's not a special ESM-Tools functionality)

<your_runscript>.yaml

```
fesom:
    foo1:
        foo2:
            foo3: bar
        foo4: [1, 2, 3]
    new_var: ${foo1}
```

choose_ is a ESM-Tools special dictionary

<your_runscript>.yaml

```
fesom:
    a_selector: True
    choose_a_selector:
        True:
            foo1: bar
    new_var: ${choose_a_selector}
```

${} – variable calling

Syntax

The value of any yaml variable can be retrieve in any file involve in the given operation (for AWI-ESM-2.1, those files are <your_runscript>.yaml, fesom-2.1.yaml, oasis3mct.yaml, echam.yaml, <machine>.yaml, <slurm/pbs>.yaml):

- Using the dot "." to call nested variables in dictionaries

<your_runscript>.yaml

```
fesom:
    foo1:
        foo2:
            foo3: bar
        foo4: [1, 2, 3]
    new_var1: ${foo1.foo2.foo3}
```

${} – variable calling

Syntax

The value of any yaml variable can be retrieve in any file involve in the given operation (for AWI-ESM-2.1, those files are <your_runscript>.yaml, fesom-2.1.yaml, oasis3mct.yaml, echam.yaml, <machine>.yaml, <slurm/pbs>.yaml):

- To add operate with variables and add together strings

<your_runscript>.yaml

```
fesom:
    a_date: DDMMYYYY
    a_file_name_with_a_date: my_file_name_${a_date}
```

<your_runscript>. yaml

```
fesom:
    unknown: 23
    question: 19
    answer: $(( ${unknown} - ${question} ))
```

More about this in the math and calendar operators chapter

${} – variable calling

Use cases

- Store math and calendar operations under a variable
- Use variables inside math and calendar operations
- Define variables to link them to namelist_changes in the configuration files so that the front-end users don't have to write the 3-level nested namelist_changes dictionary in the runscript
- Avoid conflicts in interdependent choose_ blocks
- …

More about this in the namelist_changes chapter

More about this if we get to have a look at the levante.yaml, currently in development