

● CYLC WITH ROSE ●

CYLC WORKFLOW ENGINE: <https://cylc.github.io/cylc/>
ROSE CONFIGURATION TOOLKIT: <https://metomi.github.io/rose/>

BASICS

Purpose & relationship:

- **Cylc:** a *workflow engine* for running suites (workflows) of inter-dependent applications
- **Rose:** a *toolkit* for managing the development & running of suites
- **Manage Cylc suites with Rose to:**
 - *encapsulate* required suite & task resources
 - easily edit configuration including via *GUI*
 - validate, transform & clarify with *metadata*
 - apply *version control* to suite configurations
 - access useful *built-in applications & utilities*

Sources of information & help:

- Official **documentation** (see links in heading):
 1. Cylc: *User Guide, Style Guide* + more
 2. Rose: *User Guide* including *tutorials & glossary* (both also covering Cylc), *API reference*
- **Support:** team Metomi, on-site at Met Office HQ (~D2-100) or via metomi@metoffice.gov.uk

CORE FILES & DIRECTORIES

Core files:

suite.rc	Cylc <i>suite definition</i>
rose-suite.conf	Rose <i>suite configuration</i>
rose-app.conf	Rose <i>application configuration</i>
rose-meta.conf	Rose <i>configuration metadata</i>
rose.conf	Rose <i>site & user configuration</i>
rose-suite.info	Rose (Rosie) <i>suite information</i>

Suite directory ≡ where suites are defined:

```
$HOME/roses/<suite-name>/
├─ suite.rc
├─ rose-suite.conf
├─ bin/ ...
└─ <executables>
```

Run directory ≡ where suites are run:

```
$HOME/cylc-run/<suite-name>/
├─ <processed suite dir content>
├─ <installed files>
├─ work/ ...
├─ share/ ...
└─ log/ ...
```

FILE FORMATS

Both formats below have:

- settings as key & value pairs
- section headings inside square brackets
- comments written after a hash (#) character

Cylc suite.rc format:

- also has nested & indented sub-sections
- in-line comments possible
- items written 'key = value' (with spaces)

Example: see SECTION REF

Rose Configuration Format:

- no nested sub-sections or in-line comments
- items written 'key=value' (without spaces)

Example: see SECTION REF

FURTHER SUITE.RC SYNTAX

ISO8601 datetimes for scheduling:

2000-01-01T00:00Z	returna: True
20000101T000000	returna: False
P1D	returna: True
P1Y1M1DT1H1M1S	returna: True
..... Recurrences	
P<integer>	returna: True
<isodatetime duration>	returna: True
datetime/recurrence	returna: True
offset/recurrence	returna: False
10!= 11	returna: True
.....	

Jinja2 logic for templating:

{% ... %}	statement e.g. for, if/elif/else & set
{{ ... }}	expression e.g. function, filter
{# ... #}	comment (comment-out other Jinja2)

CONFIGURATION FILE REFERENCE OVERVIEW

Intro text here.

suite.rc → uses the **suite.rc** format → Suite Definition Syntax summarises terms → *defines the suite*

[section] etc.

This	Description	Example?
+	Item	further text
*	Item	further text

rose-app.conf → uses the **Rose Configuration** format → configures *individual tasks* in a suite

Item	Item	Item	Item
%c	Item of long text with divider after	%i	Item

rose-suite.conf → uses the **Rose Configuration** format → configures the *suite as a whole*

Item	Item	Item	Item
%c	Item of long text with divider after	%i	Item

rose-meta.conf → uses the **Rose Configuration** format → *assigns metadata* to configuration settings

Item	Item	Item	Item
%c	Item of long text with divider after	%i	Item

rose-suite.info → uses the **Rose Configuration** format → specifies information *needed by Rosie*

Item	Item	Item	Item
%c	Item of long text with divider after	%i	Item

● CYLC WITH ROSE ●

CYLC WORKFLOW ENGINE: <https://cylc.github.io/cylc/>
ROSE CONFIGURATION TOOLKIT: <https://metomi.github.io/rose/>

INTERFACES: CLI (COMMAND-LINE) & GUI (GRAPHICAL USER)

Intro text here.
... etc

Section title

[section] etc.

This	Description	Example?
+	Item	further text
*	Item	further text

Formatação de Strings

Item	Item	Item	Item
%c	Item of long text with divider after	%i	Item

INTERFACES: CLI (COMMAND-LINE) & GUI (GRAPHICAL USER)

Intro text here.
... etc

Section title

[section] etc.

This	Description	Example?
+	Item	further text
*	Item	further text

Formatação de Strings

Item	Item	Item	Item
%c	Item of long text with divider after	%i	Item