

Automation Good Practices written by you

An open source project of the
Red Hat Automation CoP

Brant, Moritz, Eric, Vinny





Eric Lavarde

Principal Architect
EMEA Automation Practice
Automation CoP Manager



Moritz Schönwetter

Architect
Consulting Team Germany
AutoCoP manager, too

Agenda

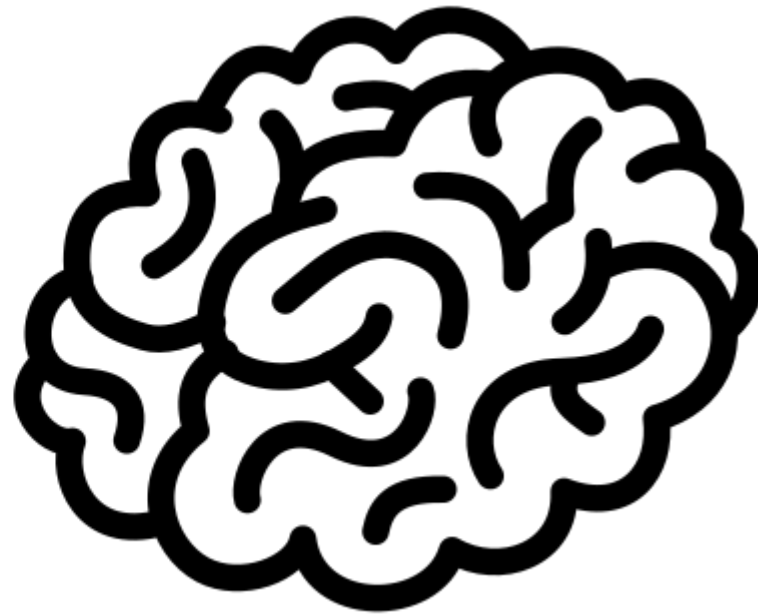
Automation Good Practices

Introduction

Examples

Call to action

Introduction



What is AGP?

What is AGP?

Exactly what it says...

Example:

5. Playbooks good practices

5.1. Keep your playbooks as simple as possible

► Details

5.2. Use either the tasks or roles section in playbooks, not both

▼ Details

Explanations

A playbook can contain `pre_tasks`, `roles`, `tasks` and `post_tasks` sections. Avoid using both `roles` and `tasks` sections, the latter possibly containing `import_role` or `include_role` tasks.

Rationale

The order of execution between `roles` and `tasks` isn't obvious, and hence mixing them should be avoided.

Examples

Either you need only static importing of roles and you can use the `roles` section, or you need dynamic inclusion and you should use *only* the `tasks` section. Of course, for very simple cases, you can just use `tasks` without `roles`.

Automation Good Practices (AGP) is a public Git repository of good practices for automation (mostly Ansible related, of course). Everybody can read and participate there.

Each good practice is made of:

- Title
- Explanation
- Rationale
- Example

History

Automation Good Practices (AGP)

First commit

February 2021, created by the Automation CoP

Based on

https://github.com/oasis-roles/meta_standards

Purpose

Provide consistent (and good) code and common structures for consultants, product teams, customers and partners

⇒ AGP is basis for linting of “validated content”

Ultimate Goal

Make a community book out of it



Source:

<https://github.com/redhat-cop/automation-good-practices/commit/0abad47b5ff7a10e1fba171254dc3cc5b81cda9f>

<https://openclipart.org/detail/259278/herodotus>

<https://ansible-lint.readthedocs.io/profiles/#production>

Maintenance process

<https://github.com/redhat-cop/automation-good-practices/>



Anybody can participate

Read the contribution guidelines, fork the repo, offer a pull request



Review process

Community based, offline review



Approval

during one of the bi-weekly Red Hat-internal Automation CoP meetings (each 2nd Wednesday)



Merge & Build

merged by one of the repo maintainers and automatically published

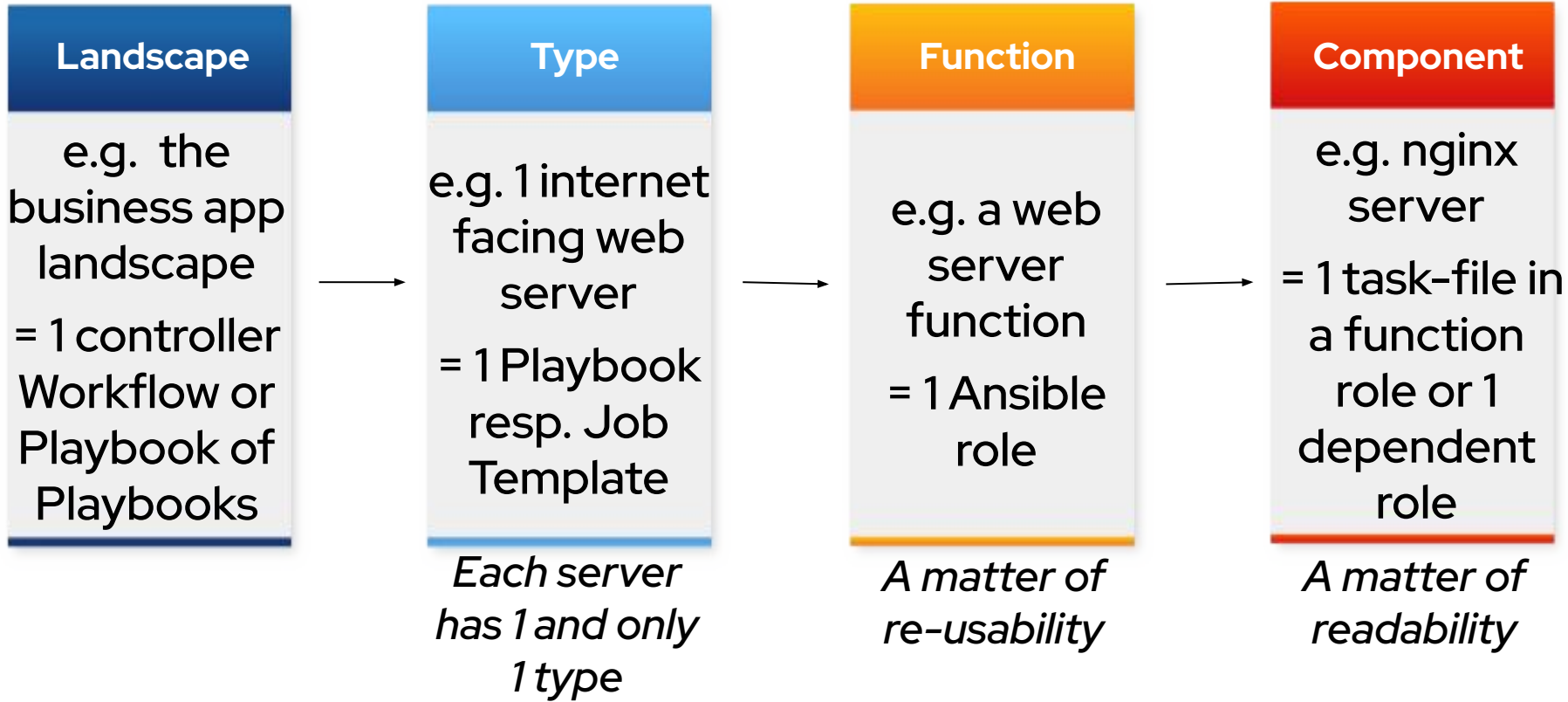
We have good practices for...

- ▶ Structures
- ▶ Roles
- ▶ Collections
- ▶ Playbooks
- ▶ Inventories
- ▶ Variables
- ▶ Plugins
- ▶ Coding in general

Few examples of good practices

The recommendations we love the most (*just like, our opinion, man*)

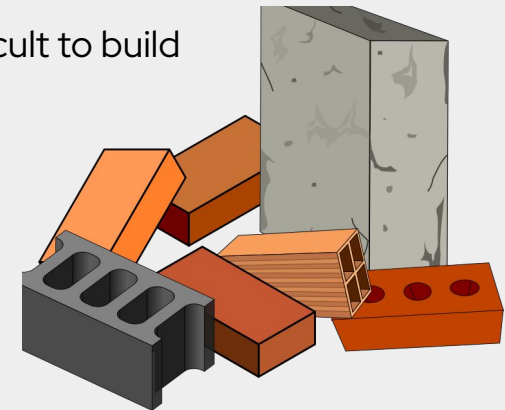




Define which structure to use for which purpose

define for which use case to use roles, playbooks, potentially workflows, and how to split the code you write.

Because a house made of bricks of various sizes is difficult to build



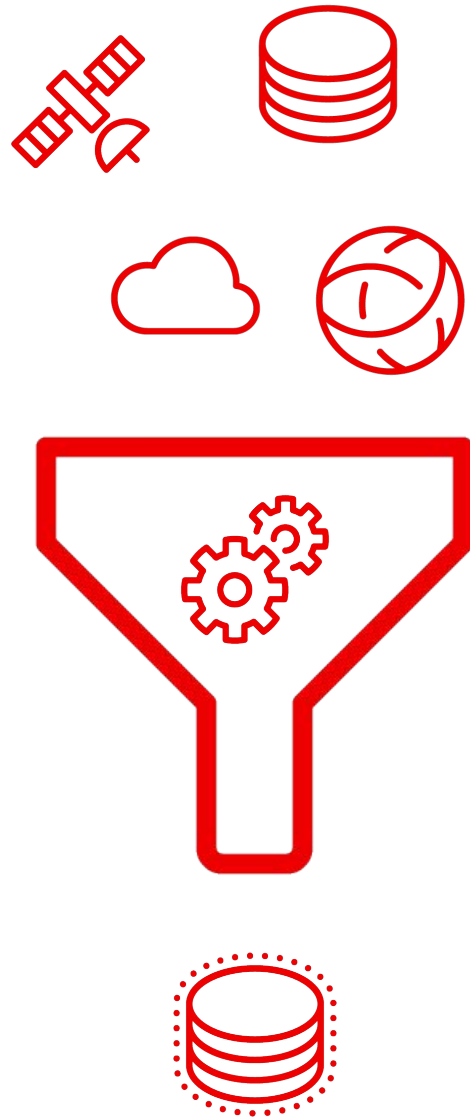


Roles - Naming parameters

Avoid names like `package` in favor of a name like `foo_package`.

Prefix internal variables with two underscores: `__foo_variable`.

Use `snake_case_naming` for variable names.



Identify your Single Source(s) of Truth and use it/them in your inventory

- Git repos
- Satellite
- Cloud inventory
- Network Infrastructure Service

Using Single Source(s) of Truth and your inventory

Some general rules (exceptions do exist)

- ▶ Filter at the source (i.e. in the API-request), not in the inventory
- ▶ Use caching
- ▶ Treat the inventory in Controller as disposable (i.e. don't define variables there)

Examples

```

inventory_example/ ①
├─ dynamic_inventory_plugin.yml ②
├─ dynamic_inventory_script.py ③
├─ groups_and_hosts ④
├─ group_vars/ ⑤
│   └─ alephs/
│       └─ capital_letter.yml
│   └─ all/
│       └─ ansible.yml
│   └─ alphas/
│       ├── capital_letter.yml
│       └─ small_caps_letter.yml
│   └─ betas/
│       └─ capital_letter.yml
│   └─ greek_letters/
│       └─ small_caps_letter.yml
│   └─ hebrew_letters/
│       └─ small_caps_letter.yml
└─ host_vars/ ⑥
    ├── host1.example.com/
    │   └─ ansible.yml
    ├── host2.example.com/
    │   └─ ansible.yml
    └─ host3.example.com/
        ├── ansible.yml
        └─ capital_letter.yml

```

```

[all]
host1.example.com
host2.example.com
host3.example.com

[alphas]
host1.example.com

[betas]
host2.example.com

[greek_letters:children]
alphas
betas

[alephs]
host3.example.com

[hebrew_letters:children]
alephs

```

(YAML is also fine here)

Define your inventory as structured directory instead of single file

- Easier to maintain and grow at scale
- Cleanly separate lists of hosts & groups ④ from the variable definitions at group ⑤ and host ⑥ level, also including dynamic plugins ② or even scripts ③
- Group and host var files named like the role they concern (capital_letter, etc...)



Split long expressions into multiple lines

>, +, -, and | are your friends

```
- name: set a very long variable
  set_fact:
    meaningless_variable: >-
      Ut ac neque sit amet turpis ullamcorper auctor.
      Cras placerat dolor non ipsum posuere malesuada at ac ipsum.
      Duis a neque fermentum nulla imperdiet blandit.
```

...your *weird* friends

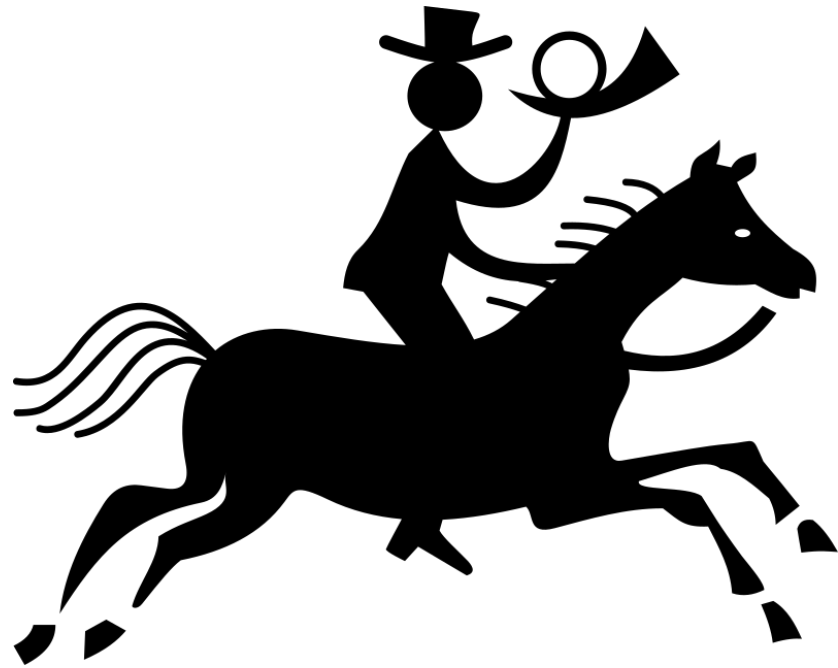
```
YAML
example: |+7\n
.....Several lines of text,\n
.....with some "quotes" of various 'types',\n
.....and also a blank line:\n
.....\n
.....and some text with\n
.....extra indentation\n
.....on the next line,\n
.....plus another line at the end.\n
.....\n
.....\n
```

And don't forget the Zen of ~~Python~~ Ansible

By *Guru* Tim Appnel



Call to action



There is a reason why
we're here!



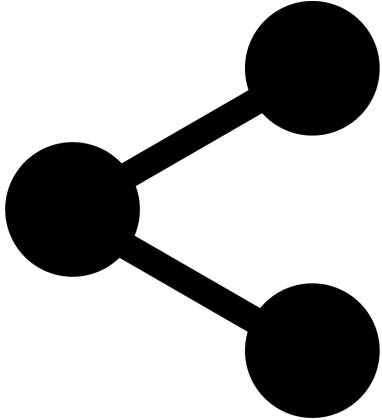
Read

It should be the beginning, shouldn't it?

<https://redhat-cop.github.io/automation-good-practices/>

What can you do?

A lot...



Apply & Share

With colleagues, customers and partners; the more people use and know about it, the better




Improve

Ask questions, create issues, address existing issues (we've got plenty of them), improve language, clarify, add new practices, create Pull Requests!

Source:
<https://redhat-cop.github.io/automation-good-practices/>
<https://github.com/redhat-cop/automation-good-practices/issues>
<https://github.com/redhat-cop/automation-good-practices/pulls>

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

 [linkedin.com/company/red-hat](https://www.linkedin.com/company/red-hat)

 [facebook.com/redhatinc](https://www.facebook.com/redhatinc)

 [youtube.com/user/RedHatVideos](https://www.youtube.com/user/RedHatVideos)

 twitter.com/RedHat