IDVOC
– YAML

-- Cyril zarak Duval, root CRI/ACU 2020

Let's talk about configuration

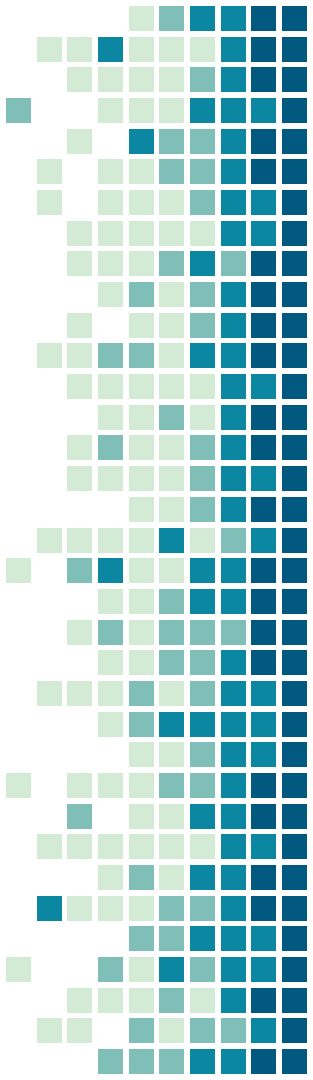
Let me introduce you to a new job:
YAML engineer

Configuration

- Article 3 of the 12 factors
  - strict separation of config from code
- What does it means ?
- Softwares must not include configuration within the code
- Configuration shall not even be in the code VCS
- Configuration will be different between dev, staging and prod(s)

What is configuration ?

- What is configuration ?
  - Everything removing genericity
  - Everything that can change from an env to another
- Examples:
  - Address of the database
  - IP/port to bind to
  - Log-level
  - Path(s) to store data
  - ....



How to provide config ?

- How to provide configuration ?
  - env variables
    - Recommended for simple cases, but limited
  - Config files
    - A bit more complex to setup/provide but advanced
- How to write a config file ?
- Introducing to the most well known format in DevOps: YAML

YAML

- YAML = Yet Another Markup Language
- Used for docker-compose, gitlab-ci, kubernetes, helm charts, ansible, elasticsearch, argo, harbor, ...
- A superset of JSON to make it more human readable
  - Meaning that JSON is valid YAML
  - Even small JSON snippets embedded within a YAML file
- .yml or .yaml extension
- Libs in every language to parse it

YAML - the simple way

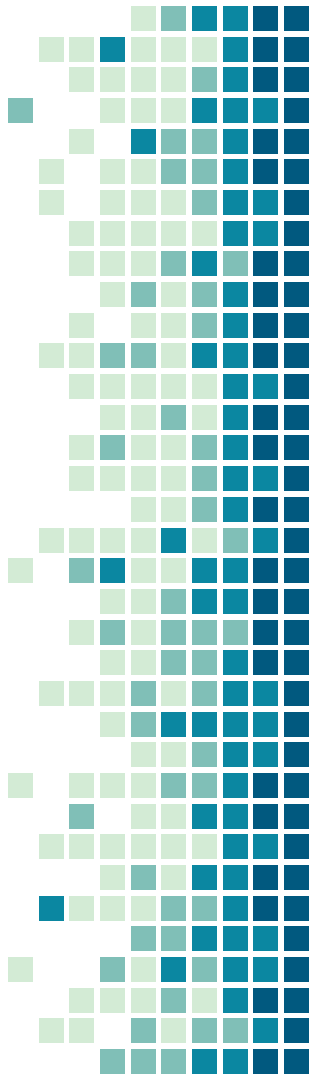
- YAML is a key-value format
- key is a string, value can be :
  - string
  - int
  - boolean
  - list
  - mapping
- Indentation is important
- Quoting can be used

YAML – the simple way

```yaml
---

key_1: value
key_2: 140
key_3: 1.0
key_4: "1.0"
key_5:
    - 0
    - 1
    - 2
key_6:
    subkey_1: value with a "quote"
    subkey_2: '"full quoted"'
    subkey_3:
        - subsubkey_1: value1
          subsubkey_2: value2
        - subsubkey_1: value3
          subsubkey_2: value4
key_7:
    bool_1: True
    bool_2: False
    bool_3: yes
    bool_4: no
    bool_5: on
    bool_6: off
```

YAML - Advanced features

```yaml
---
common: &common
  toto: {"key": "value"}
  titi: tutu

a:
  - <<: *common
  - <<: *common
    titi: something else

key: |
    multi
    line
    value

key2: |-
    "another way to escape quotes

key3: |
    2

key4: >
    lorem
    ipsum
    sit

            amet dolor
```



```yaml
---
common: &common
  toto: {"key": "value"}
  titi: tutu

a:
  - <<: *common
  - <<: *common
    titi: something else

key: |
    multi
    line
    value

key2: |-
    "another way to escape quotes

key3: |
    2

key4: >
    lorem
    ipsum
    sit

            amet dolor
```

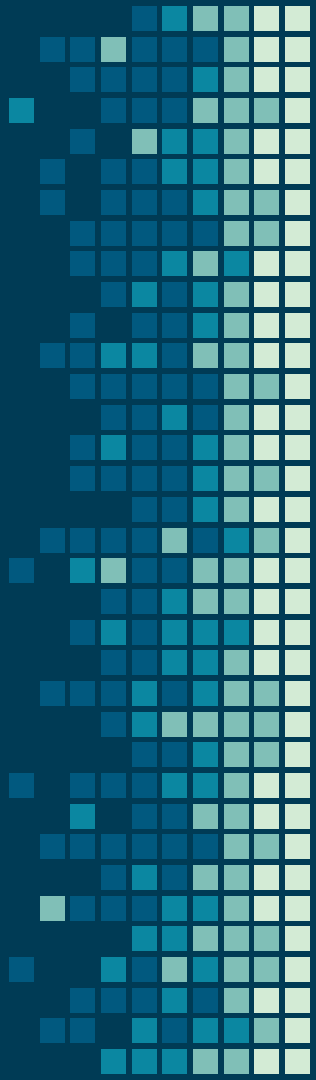
```
$ cat /tmp/file.yaml | yq
{
  "common": {
    "toto": {
      "key": "value"
    },
    "titi": "tutu"
  },
  "a": [
    {
      "toto": {
        "key": "value"
      },
      "titi": "tutu"
    },
    {
      "toto": {
        "key": "value"
      },
      "titi": "something else"
    }
  ],
  "key": "multi\nline\nvalue\n",
  "key2": "\"another way to escape quotes",
  "key3": "2\n",
  "key4": "lorem ipsum sit\n\n amet dolor\n"
}
```

YAML - tools

- ➔ yq is a useful tool to parse yaml in bash
- ➔ yamllint detects linting errors, inconsistencies and warn you about possible misuage

Thanks !

Questions ?



Slides available on zarak.fr/

Contact: cyril@cri.epita.fr

zarak production#5492