




# DevNet Experts.

Topic- YAML

 +91 9892028199

 [devnetexperts@gmail.com](mailto:devnetexperts@gmail.com)





# OVERVIEW

- What is YAML?
- Why YAML?
- Understanding YAML Document



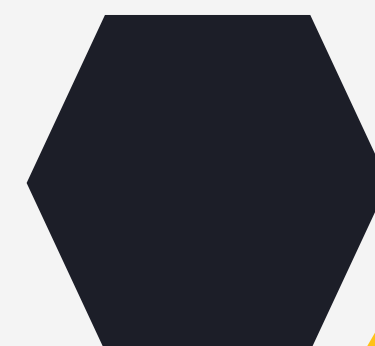
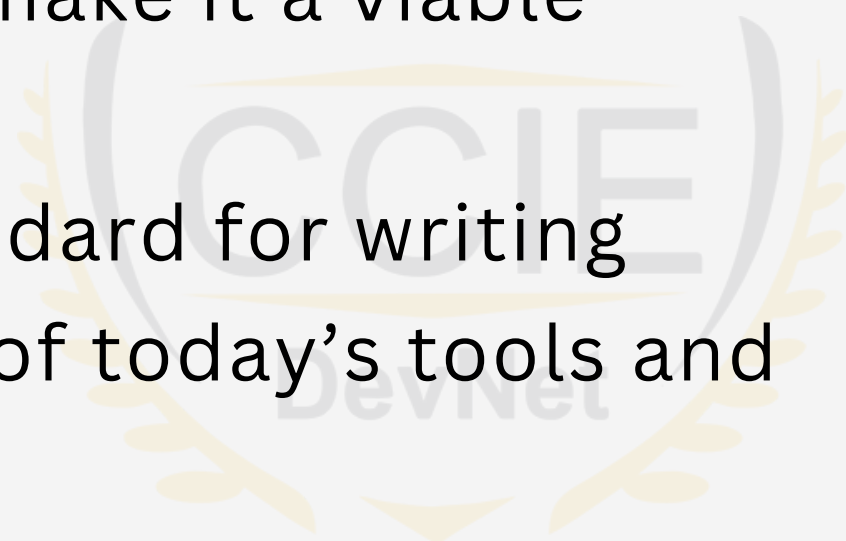
# What is YMAL?

- YAML originally started as Yet Another Markup Language
- It's original name was inspired by XML which stands for eXtended Markup Language
- However YAML is just a data interchange format similar to JSON and not a markup language as such
- Hence it rechristened itself later as YAML Ain't Markup Language



# Why YAML?

- YAML has increased steadily in its popularity resulting in its widespread usage
- Its serialization capabilities make it a viable replacement for JSON
- It has become a defacto standard for writing configuration files for many of today's tools and frameworks
- It has broad language support and maps easily into native data structures
- It is easy to read and write, hence the language of choice for writing config files



# Understanding YAML Document.

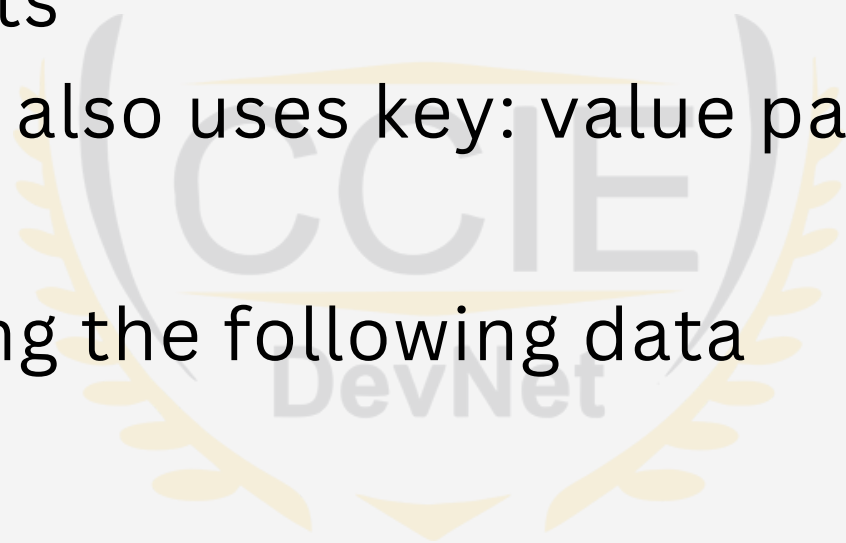
```
---  
- name: Update web servers  
  hosts: webservers  
  remote_user: root  
  
  tasks:  
  - name: Ensure apache is at the latest version  
    ansible.builtin.yum:  
      name: httpd  
      state: latest  
  
- name: Update db servers  
  hosts: databases  
  remote_user: root  
  
  tasks:  
  - name: Ensure postgresql is at the latest version  
    ansible.builtin.yum:  
      name: postgresql  
      state: latest
```



# Understanding YAML Document.

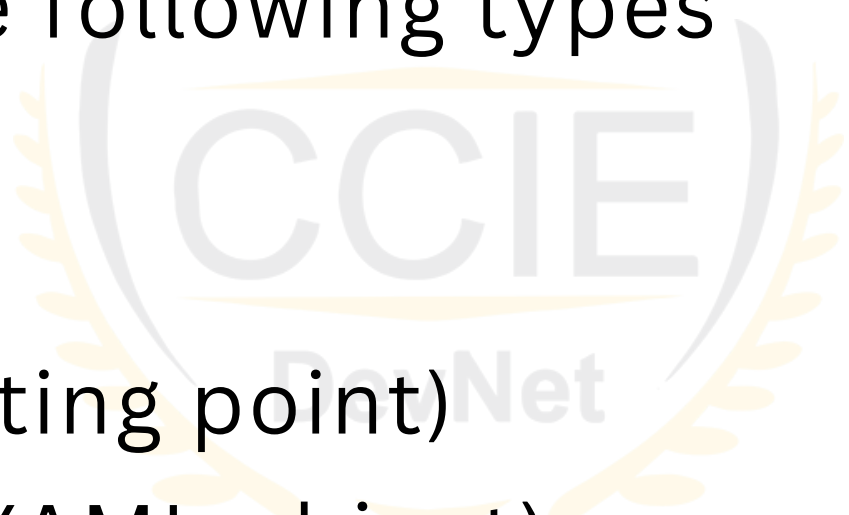
- The three dashes on top indicate the start of YAML document
- A YAML file can consist of multiple documents each starting with 3 dashes
- And optionally ending with 3 dots
- Similar to values in JSON, YAML also uses key: value pairs to specify data
- YAML supports storing data using the following data structures

1. Scalars - simple key - value pairs
2. Lists - Sequence of key - value pairs
3. Dictionaries - Nested combination of scalars or lists (hashes or maps)



# YAML Data Types.

- YAML stores data as a combination of key, value pair
- Value can be one of the following types
  1. A string
  2. A number (integer, floating point)
  3. An dictionary (Nested YAML object)
  4. An list
  5. A boolean

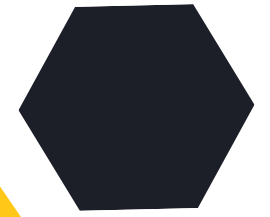
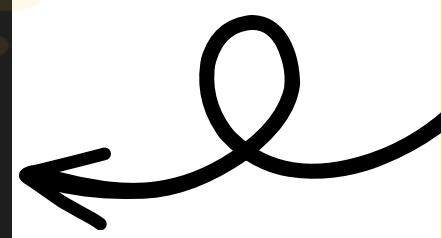


# YAML Dictionaries

- YAML dictionary is a nested representation consisting of:
  1. Scalars
  2. Lists
  3. Dictionaries

```
---
networks:
- gateway: 11.117.0.1
  pool: [11.117.0.5 to 11.117.0.254]
  segments: [tenant]
  subnet: 11.117.0.0/24
  vlan_id: 17
- gateway: 11.118.0.1
  pool: [11.118.0.5 to 11.118.0.254]
  segments: [storage]
  subnet: 11.118.0.0/24
  vlan_id: None
- segments: [external]
  vlan_id: 400
- segments: [provider]
  vlan_id: None
```

- Defining networks using the key networks.
- Value is a list of gateways and segments.
- Each value is combination of lists and scalars





# YAML List.

- YAML Lists support storing a sequence of values
- Lists can consist of scalars, lists or dictionaries

```
---
router:
  bgp:
    - id: 100
      bgp:
        router_id: 1.1.1.1
        fast_external_fallover: null
        update_delay: 15
      neighbor:
        - id: 2.2.2.2
          remote_as: 200
        - id: 3.3.3.3
          remote_as: 300
```



List of BGP neighbors

# Parsing YAML with Python.

- Python has an inbuilt package called `pyyaml` to work with YAML files
- We first open YAML file with the Python with `construct`
- We can then load YAML data into Python using `safe_load` function
- YAML data will then be available as Python dictionary



# Parsing YAML with Python. (cont.)

```
parse_yaml.py
1  import yaml
2  import pprint
3  pp = pprint.PrettyPrinter(indent=2)
4
5  with open ("test.yaml") as yaml_file:
6      network_data = yaml.safe_load(yaml_file)
7      pp.pprint(network_data)
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
• nexadmin@DESKTOP-89IJ1T7:~/work/temp$ python parse_yaml.py
{ 'networks': [ { 'gateway': '11.117.0.1',
                  'pool': ['11.117.0.5 to 11.117.0.254'],
                  'segments': ['tenant'],
                  'subnet': '11.117.0.0/24',
                  'vlan_id': 17},
                { 'gateway': '11.118.0.1',
                  'pool': ['11.118.0.5 to 11.118.0.254'],
                  'segments': ['storage'],
                  'subnet': '11.118.0.0/24',
                  'vlan_id': 'None'},
                {'segments': ['external'], 'vlan_id': 400},
                {'segments': ['provider'], 'vlan_id': 'None'}] }
```

```
○ nexadmin@DESKTOP-89IJ1T7:~/work/temp$
```





DEMO.