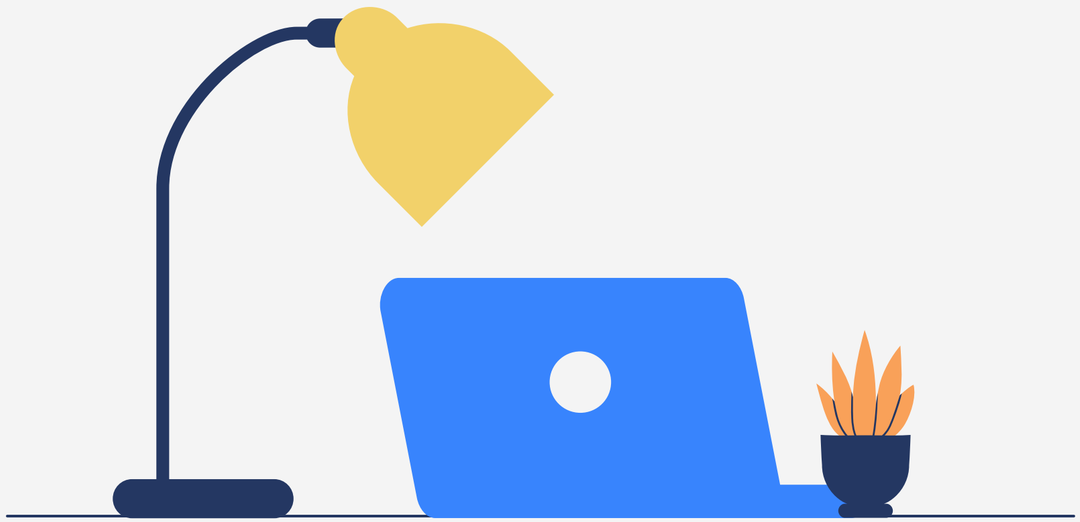
YAML

CRASH COURSE

By Sandip Das

Topics

Let's learn YAML easy way



What is YAML?

And what are the use cases ?

How to write a YAML file?

and how to configure the development environment for it?

What are the YAML data types ?

and when to use which data type?

How it's different from JSON?

explain with hands-on demo

Real life use cases examples

With Kubernetes, GitHub Actions, AWS CodeBuild & Code Deploy and more

How to use Python to read a YAML file?

with a very simple to understand Python script

What is YAML ?

YAML stand's for **YAML Ain't Markup Language,** is a serialization language that has steadily increased in popularity over the last few years. It's often used as a format for configuration files, but its object serialization abilities make it a viable replacement for languages like JSON.

YAML has varieties of language support and maps easily into native data structures. It's also easy for humans to read, which is why it's a good choice for configuration. The YAML acronym was shorthand for Yet Another Markup Language. But the maintainers renamed it to YAML Ain't Markup Language to place more emphasis on its data-oriented features.
YAML is a superset of JSON, we can also write JSON-style maps and sequences.

Use cases :

YAML is used as the configuration file for many software/systems, mainly used for DevOps tools such as:

Kubernetes configuration files, GitHub Actions Workflow file, Ansible Playbook, AWS CodeBuild & CodeCommit configuration file, and much more such use cases.

```yaml
--- # start of YAML document
# - is used for comment
#string
name: "Sandip Das"
#nuber
name: 30
# string without quotes
city: Kolkata
#array
skills:
  - DevOps
  - Cloud
  - Programming
  - Mentoring
  - Communication
#object / nested object
cars:
  - model:
      colour: Red
      name: "Maruti Dzire"
# Boolean: we can use: True/False or Yes/No
is_married : Yes
has_child : True
# Since YAML is a superset of JSON,
# We can also write JSON-style maps and
# sequences:
json_map: {"key": "value"}
json_seq: [3, 2, 1, "Yessss..working"]
```

How to Write YAML File?

and how to configure Development environment for it?

Point to note before writing

- YAML is a data serialization language designed to be directly writable and readable by humans
- Check https://yaml.org for the latest recommendations
- YAML is CASE sensitive
- End your YAML file with the .yaml or .yml extension
- YAML is a superset of JSON

Dev Environment Setup

- Best Editors/IDE: VS Code, Atom
- Plugins: YAML plugins or language support or lining available in all major IDEs, before start writing, should install the relevant plugin

Basic Syntax

- **The file starts with ---**
- We can write configuration blocks in the YAML in 2 ways:

Indented Block and Inline Block e.g.

```
--- # Indented Block
name: Sandip Das
age: 30
--- # Inline Block
{name: Sandip Das, age: 30}
```

- **We can define value like:**

```
---
value
```

- **named value:**

```
sample_key: sample_value
```

- **List:**

```
- 1
- 2
- 3
- 'sample string'
```

Named list:

```
---
sample_list:
- 1
- 2
- 3
- 'sample string'
```

Dictionary:

```
key1: val1
key2: val2
key3: val3
```

Named Dictionary:

```
sample_dict:
  key1: val1
  key2: val2
  key3: val3
```

Dictionary in flow style:

```
sample_dict: {key1: val1, key2: val2, key3: val3}
```

List of Dictionaries:

```
developers:
- name: Sandip Das
  age: 30
- name: James Smith
  age: 54
```

Alternative list : (Flow style)

```
sample_list: [1, 2,3,  'a sample string']
```

YAML Data Types

Supported Basic Data Types

```
a: 1        # integer
a: 1.234    # float
b: 'abc'    # string
b: "abc"
b: abc
c: false    # boolean type
d: 2015-04-05   # date type
```

YAML Anchors and Alias

We can place Anchors (&) on an entity to mark a multi-line section. We can then use an Alias (*) call that anchor later in the document to reference that section. E.g.

```
some_var: &someVarRef Test Value
some_var_value_referning: *someVarRef
```

Advance Data Types

Array: Either like below or: ["DevOps", "Cloud", "Programming"...]

```
- DevOps
- Cloud
- Programming
- Mentoring
- Communication
```

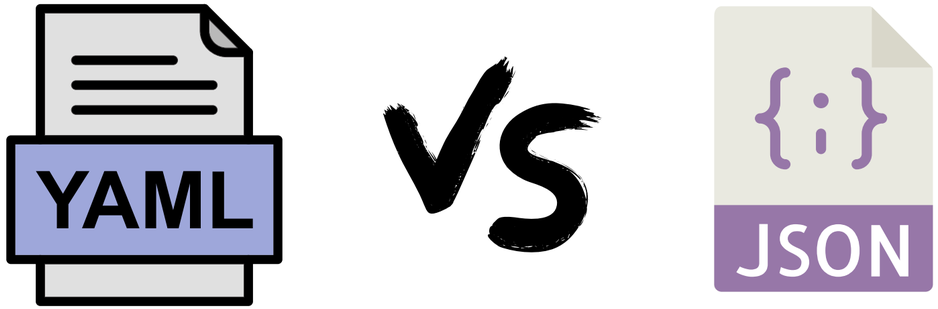
Array of Objects:

```
cars:
 - name: "Maruti Dzire"
   colour: Red
 - name: "Nexon EV"
   colour: Dark
```

Nested Array of Objects:

```
cars:
 - Model
      name: "Maruti Dzire"
      colour: Red
 - Model:
      name: "Nexon EV"
      colour: Dark
```

How YAML is different from JSON



What is JSON?

JSON stands for **JavaScript Object Notation**. It is a lightweight format for storing and transporting data, often used when data is sent from a server to a web page or server to server or program to program communication, works as a common standard data format. It is "self-describing" and easy to understand

JSON Example:

```
{
 "name": "Sandip Das",
  "age": 30,
 "city": "Kolkata",
 "skills": [
 "DevOps",
 "Cloud",
 "Programming",
 "Mentoring",
 "Communication"
 ]
}
```

YAML	JSON
Comments are denoted with a hash or number sign.	Comments are not allowed at all.
Hierarchy is denoted by using double space characters. Tab characters are not allowed.	Objects and Arrays are denoted in braces and brackets.
String quotes are optional unless using special character but it supports single and double quotes.	Strings must be in double quotes.
Root node can be any of the valid data types.	Root node must either be an array or an object.

From a User point of view, YAML is more human readable and editable than JSON

Yaml Real life use cases

Kubernetes

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx:1.7.9
    ports:
    - containerPort: 80
```

Appspec.yml

```yaml
version: 0.0
os: linux
files:
  - source: /
    destination: /home/ubuntu/app
hooks:
  ApplicationStop:
    - location: deployment_scripts/stop_server.sh
      timeout: 300
      runas: root

  BeforeInstall:
    - location: deployment_scripts/before_install.sh
      timeout: 300
      runas: root

  AfterInstall:
    - location: deployment_scripts/after_install.sh
      timeout: 300
      runas: root

  ApplicationStart:
    - location: deployment_scripts/start_server.sh
      timeout: 300
      runas: root

  ValidateService:
    - location: deployment_scripts/validate_service.sh
      timeout: 300
      runas: root
```

AWS Codebuild BuildSpec.yml

```yaml
version: 0.2
run-as: root


phases:
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --username AWS --password-stdin $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGIO
      - REPOSITORY_URI=$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $IMAGE_REPO_NAME:$IMAGE_TAG .
      - docker tag $IMAGE_REPO_NAME:$IMAGE_TAG $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - docker push $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
      - echo Writing image definitions file...
      - printf '[{"name":"%s","imageUri":"%s"}]' $CONTAINER_NAME $REPOSITORY_URI:$IMAGE_TAG > imagedefinitions.json
artifacts:
    files: imagedefinitions.json
```

Ansible Playbook

```yaml
- name: Installl and Verify apache installation
  hosts: appservers
  vars:
    http_port: 80
    max_clients: 200
  remote_user: ubuntu
  become: yes
  become_method: sudo
  tasks:
  - name: Ensure apache is at the latest version
    ansible.builtin.apt:
      name: apache2
      state: latest

  - name: Ensure apache is running
    ansible.builtin.service:
      name: apache2
      state: started
```

github-actions-demo.yml

```yaml
name: GitHub Actions Demo
on: [push]
jobs:
  Explore-GitHub-Actions:
    runs-on: ubuntu-latest
    steps:
      - run: echo "🎉 The job was automatically triggered by a ${{ github.event_name }} event."
      - run: echo "🐧 This job is now running on a ${{ runner.os }} server hosted by GitHub!"
      - run: echo "🔎 The name of your branch is ${{ github.ref }} and your repository is ${{ github.repository }}."
      - name: Check out repository code
        uses: actions/checkout@v2
      - run: echo "💡 The ${{ github.repository }} repository has been cloned to the runner."
      - run: echo "🖥️ The workflow is now ready to test your code on the runner."
      - name: List files in the repository
        run: |
          ls ${{ github.workspace }}
      - run: echo "🍏 This job's status is ${{ job.status }}."
```

Read any YAML file using Python

Python Code

```
import yaml

if __name__ == '__main__':
 stream = open("sample.yaml", 'r')
 dictionary = yaml.load(stream, Loader=yaml.FullLoader)
 try:
  for key, value in dictionary.items():
   print (key + " : " + str(value))
 except Exception as e:
  print("e",e)
```

Steps:

- **import python inbuild yaml module**
- **open any yaml file in read mode**
- **use yaml.load() function to convert yaml readable file stream data into python dictionary**
- **We can use keys / values of the dictionary variable as needed**

Useful Links

- Demo Code Repo
- Official YAML site with Official YAML Module/Plugin for all programming languages
- Ansible playbook demo code
- Kubernetes Configuration Demo Codes/Templates
- Online YAML Validator

SUBSCRIBE

http:// Learn.sandipdas.in

Contact Me

contact@sandipdas.in

FOLLOW

Support My Work

Via **Patreon**: https://www.patreon.com/learnwithsandip

Via **Buy Me a Coffee**: https://www.buymeacoffee.com/LearnWSandip