➢ **YAML:**

- YAML **Ain't Markup Language**.
- YAML is a **human-readable data serialization language** that is often used for writing configuration files.
- AML is a popular programming language because it is designed to be easy to read and understand.
- It can also be used in conjunction with other programming languages. Because of its flexibility, and accessibility, YAML is used by Ansible to create **automation processes**, in the form of **Ansible Playbooks**.

BENEFITS:

- Human Readable
- Simple and Clean Syntax
- Easy to implement and use
- Matches data structures of programming languages
- Consistent data model
- Single pass processing
- Expressive and Extensible

USE CASES:

- Configuration files
- Log files
- Inter-process messaging
- Cross-language data sharing
- Object persistence
- Complex data structures

WHY YOU SHOULD USE YAML:

- Easily readable by humans, YAML files are expressive and extensible.
- They are easy to implement and use.
- They are easily portable between programming languages.
- They match the native data structures of agile languages.

- YAML files have a consistent model to support generic tools.
- They support one-pass processing.
- Convenient to use, so you no longer need to add all of our parameters to the command line.
- Maintenance can be done – YAML files can be added to the source control to track the changes.
- **Flexibility:** You can create much complex structures using YAML than you can use on command line.

❖ YAML, JSON & XML:

- YAML and JSON share similarities in data representation, however, YAML stands out for its readability, expressiveness and support for complex data structures. YAML is a superset of JSON, meaning that it contains all the features of JSON in addition to expanded features and commands.
- JSON (JavaScript Object Notation) utilizes a more explicit syntax with braces {}, brackets [], and commas. While concise and widely used, JSON's syntax might become less readable, especially in larger datasets. JSON's support for data structures is comparatively limited, primarily featuring arrays, objects and scalar values.
- When compared to XML, YAML offers a more concise and human-friendly alternative, emphasizing simplicity and ease of comprehension in data representation and exchange.
- YAML and XML have fundamental differences in syntax and purpose.

XML	JSON	YAML

`<Servers>`
  `<Server>`
    `<name>Server1</name>`
    `<owner>John</owner>`
    `<created>123456</created>`
    `<status>active</status>`
  `</Server>`
`</Servers>`
	`{`
	  `Servers: [`
	    `{`
	      `name: Server1,`
	      `owner: John,`
	      `created: 123456,`
	      `status: active`
	    `}`
	  `]`
	`}`
		`Servers:`
		  `-` `name: Server1`
		     `owner: John`
		     `created: 123456`
		     `status: active`

XML:

- It is a pioneer in many domains. XML was originally designed to be backwards compatible with the **Standard Generalized Markup Language (SGML),** which was designed to support **structured documentation.**

JSON:

- It designs goal is simplicity and universality, with its goal to generate and parse. It has reduced human readability, but its data can be processed easily by every **modern programming environment**.

YAML:

- It designs goals are human readability and a more complete information model. YAML is more complex to generate and parse, therefore it can be viewed as a natural superset of JSON.

  NOTE: Every JSON file is also a valid YAML file.

❖ YAML AND DEVOPS:

- YAML plays a pivotal role in DevOps and is instrumental in automation, orchestration and configuration management. Within DevOps practices, YAML files serve as blueprints to define sequences of actions and configurations in an easily understandable format.
- YAML is used to define **infrastructure as code,** which is the use of code, rather than manual processes, to define and manage IT infrastructure.
- YAML is used to create **deployment files** for applications that specify app configurations, dependencies, resource limits and other information important to efficient application deployment and performance.
- YAML files help reduce **deployment errors** and increase application delivery speed through version-control and automation.
- YAML can play an important role in **continuous integration and continuous delivery (CI/CD) pipelines,** an important **agile DevOps workflow.** Similar to infrastructure configuration and deployments, YAML files are used to define the pipeline steps and targets, and ultimately help automate CI/CD process.

❖ YAML AND DEVOPS TOOLS:

- Many tools and programs used by DevOps teams leverage YAML, including:
    - Ansible   - GitHub
    - Kubernetes   - Docker Compose

❖ YAML SYNTAX:

- YAML files use a **.yml** or **.yaml** extension, and follow specific syntax rules.
- YAML has features that come from Perl, C, XML, HTML, and other programming languages. YAML is also a superset of JSON, so JSON files are valid in YAML.
- There are no usual format symbols, such as braces, square brackets, closing tags, or quotation marks.
- YAML files are simpler to read as they use **Python-style** indentation to determine the structure and indicate nesting.
- **Tab characters** are not allowed by design, to maintain portability across systems, so whitespaces—literal space characters—are used instead.
- Comments can be identified with a pound or **hash symbol (#).**
- YAML does not support **multi-line comment,** each line needs to be suffixed with the pound character.
- Three **dashes (---)** are used to signal the **start of a document,** while each document ends with **three dots (...).**

BASIC EXAMPLE OF A YAML FILE:

```
#Comment: This is a supermarket list using YAML
#Note that - character represents the list
---
food:
   - vegetables: tomatoes #first list item
   - fruits: #second list item
        citrics: oranges
        tropical: bananas
        nuts: peanuts
        sweets: raisins
```

❖ Y AML SYNTAX EXAMPLE:

- Here's an example of a simple YAML file for a student record that demonstrates the syntax rules:

```
#Comment: Student record
#Describes some characteristics and preferences
---
name: Martin D'vloper #key-value
age: 26
hobbies:
  - painting #first list item
  - playing_music #second list item
  - cooking #third list item
programming_languages:
  java: Intermediate
  python: Advanced
  javascript: Beginner
favorite_food:
  - vegetables: tomatoes
  - fruits:
      citrics: oranges
      tropical: bananas
      nuts: peanuts
      sweets: raisins
```

❖ STRUCTURE OF A YAML FILE:

Key Value Pair	Array/Lists	Dictionary/Map

Fruit: Apple
Vegetable: Radish
Liquid: Water
Meat: Goat
	Fruits:
	  - Orange
	  - Banana
	  - Mango
	Vegetables:
	  - Potato
	  - Tomato
	  - Carrot
		Banana:
		  Calories: 200
		  Fat: 0.5g
		  Carbs: 30g
		Grapes:
		  Calories: 100
		  Fat: 0.4g
		  Carbs: 20g

KEY VALUE PAIR:

- The basic entry in a YAML file is a **key value pair**. After the **Key** and colon there is a space and then the **value**.

  Fruit: Apple
  Liquid: Water

ARRAYS/LISTS:

- Lists would have a number of items listed under the name of the list. The elements of the list would start with a **'-'**.

  Fruits:
  - Orange
  - Banana
    Vegetables:
    - Potato
    - Tomato

DICTIONARY/MAP:

- A more complex type of YAML file would be a **Dictionary and Map**.

  Banana:
    Calories: 200
    Fat: 0.5g
  Grapes:
    Calories: 100
    Fat: 0.4g