

# Kubernetes – YAML Explained

**VISHWANATH M S**

**[VISHWACLOUDLAB.ORG](https://vishwacloudlab.org)**

# What is YAML?

YAML, which stands for Yet Another Markup Language or  
YAML Ain't Markup Language (depending who you ask)

Using YAML for K8s definitions gives you a number of advantages, including:

- **Convenience:** You'll no longer have to add all of your parameters to the command line
- **Maintenance:** YAML files can be added to source control, so you can track changes
- **Flexibility:** You'll be able to create much more complex structures using YAML than you can on the command line

# Type of Structures in YAML

There are only **two** types of structures you need to know about in YAML:

- Lists
- Maps

That's it. You might have maps of lists and lists of maps and so on.....

# YAML Maps

---

apiVersion: v1

Kind: pod

---

→ Called as separator

used when multiple structures are defined

Next,

There are 2 pre-defined values "v1" and "pod" mapped to two keys, apiVersion and kind.

# YAML Maps Continued ...

More complicated structures

Creating a key that **maps** to another **map**

```
---  
apiVersion: v1  
kind: Pod  
metadata:  
  name: rss-site  
  labels:  
    app: web
```

Indenting the lines are important

Min is “1” space

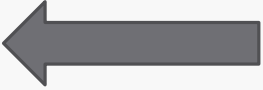
Indenting should be **CONSISTENT**

**\*\*\*NEVER USE TABS in a YAML file\*\*\***

# YAML Lists

YAML lists are literally a sequence of objects.

args:

- sleep
  - "3000"
  - message
  - "Kube is Beautiful"
- 
- Items in the list**

# YAML Maps in List

containers:

- name: front-end

image: nginx

ports:

- containerPort: 80



**Maps in the list**



**Maps in the list  
Containing Key:Value**

# Review on YAML

- Maps, which are groups of name-value pairs
- Lists, which are individual items
- Maps of maps
- Maps of lists
- Lists of lists
- Lists of maps



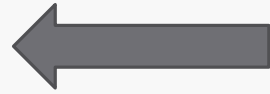
# Creating a POD using YAML

```
---
apiVersion: v1
kind: Pod
metadata:
  name: rss-site
  labels:
    app: web
spec:
  containers:
    - name: front-end
      image: nginx
      ports:
        - containerPort: 80
    - name: rss-reader
      image: nickchase/rss-php-nginx:v1
      ports:
        - containerPort: 88
```

# Let's look at each piece closer

---

apiVersion: v1



**We start with this**  
**Currently we use v1.**

kind: Pod

metadata:

name: rss-site

labels:

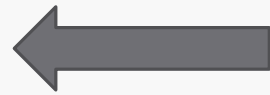
app: web

# Let's look at each piece closer

---

apiVersion: v1

kind: Pod



**Object type to be created.  
Specifying to create a POD.**

metadata:

name: rss-site

labels:

app: web

**Also, we could create**

- Deployment
- Job
- Service

# Let's look at each piece closer

---

apiVersion: v1

kind: Pod

metadata: 

name: rss-site

labels:

app: web

**Name of the Pod**

**Label for Identifying POD to Kubernetes**

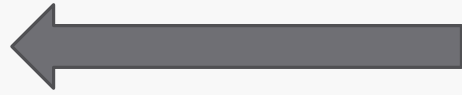
We Could have anykind of **Key:Value** pair for labels.

**Used as label selector for calling the POD and also for filtering the POD's**

# Let's look at each piece closer

.....

spec:



containers:

- name: front-end  
image: nginx  
ports:
  - containerPort: 80
- name: rss-reader

.....

**Actual Objects that  
make up the POD.**

**Spec, property can  
include**

- Containers**
- Storage Volumes**

# Let's look at each piece closer

.....

spec:

containers:

– name: front-end

image: nginx

ports:

– containerPort: 80

– name: rss-reader

.....

This is the name given to the container that we are trying to create.

This is the name of the image which we are trying to pull from the Docker or internal registry of images.

# Let's look at each piece closer

.....

spec:

containers:

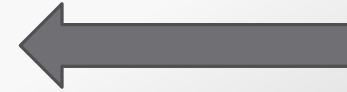
- name: front-end

image: nginx

ports:

- containerPort: 80

- name: rss-reader



The port number  
exposed to access the  
container

.....

# kubectl commands

# To run the yml file on the Kube Cluster.

**\$ kubectl apply -k <yaml filename>**

# To check the pod status

**\$ kubectl get pods**



