# **Tutorial: Understanding Kubernetes Objects**

## 

[**Tutorial: Understanding Kubernetes Objects**](#_kjndtfqvjipi) **1**

[Introduction](#_5fq572u13lw1) 1

[1. Kubernetes Object Basics](#_ekl6jvjeccdy) 2

[2. YAML Primer for Kubernetes](#_euza3ez8xckv) 2

[Key YAML Rules:](#_qo8f6xk3xa1h) 2

[Sample YAML Structure](#_u7ke0z2fe3q6) 2

[3. Kubernetes Objects (Beginner to Advanced)](#_uhh88b8r430h) 2

[1. Pod](#_eqknjmesufeq) 2

[2. Deployment](#_yv5dbsthnj8) 3

[3. ReplicaSet](#_6m9lp3ad5zfm) 4

[4. Service](#_yipeslb1huyy) 4

[5. ConfigMap](#_8r2q8cqdiy4x) 5

[6. Secret](#_x1e0rxuyylsq) 5

[7. PersistentVolume (PV) and PersistentVolumeClaim (PVC)](#_xwi0jmmgayah) 6

[8. StatefulSet](#_oofnkfh3f4qy) 6

[9. DaemonSet](#_wck8ybwb686f) 7

[10. Job and CronJob](#_r17npu8dwv) 8

[11. Ingress](#_3kfdybbrdco1) 8

[12. CustomResourceDefinition (CRD)](#_6eh8cxn9k7mc) 9

[4. Summary](#_a4dz6fpslnt7) 10

## 

## **Introduction**

Kubernetes is built around a set of API objects that define the desired state of the system, such as running applications, associated networking, and storage resources. This guide will walk you through Kubernetes objects, from basic to advanced, including their purpose, key fields, and YAML file structures.

We’ll start with fundamental concepts like Pods and progress to complex objects like StatefulSets and CustomResourceDefinitions (CRDs). By the end, you’ll also learn the importance of YAML and its correct usage for defining Kubernetes objects.

## **1. Kubernetes Object Basics**

A Kubernetes object is a "record of intent"—once created, Kubernetes works to maintain it in the specified state. Each object has:

* **Metadata:** Information like name, namespace, and labels.
* **Spec:** Desired state of the object.
* **Status:** Current state of the object, updated by Kubernetes.

## **2. YAML Primer for Kubernetes**

YAML is the primary format for defining Kubernetes objects. It uses indentation to structure data hierarchies, and incorrect indentation can lead to errors.

### **Key YAML Rules:**

* Use **spaces, not tabs** for indentation.
* Indentation typically uses 2 spaces.
* Proper nesting is crucial for hierarchical data.

### **Sample YAML Structure**

apiVersion: v1

kind: Pod

metadata:

name: my-pod

spec:

containers:

- name: my-container

image: nginx

## **3. Kubernetes Objects (Beginner to Advanced)**

### **1. Pod**

* **Description:** The smallest deployable unit in Kubernetes, encapsulating one or more containers.
* **Key Fields:** containers, volumes.

**Example:** apiVersion: v1

kind: Pod

metadata:

name: my-pod

spec:

containers:

- name: nginx-container

image: nginx:latest

ports:

- containerPort: 80

### **2. Deployment**

* **Description:** Manages Pods with declarative updates, ensuring desired replicas are running.
* **Key Fields:** replicas, selector, template.

**Example:** apiVersion: apps/v1

kind: Deployment

metadata:

name: my-deployment

spec:

replicas: 3

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx-container

image: nginx:latest

### **3. ReplicaSet**

* **Description:** Ensures a specified number of Pod replicas are running. Usually managed by Deployments.
* **Key Fields:** replicas, selector, template.

**Example:** apiVersion: apps/v1

kind: ReplicaSet

metadata:

name: my-replicaset

spec:

replicas: 2

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx-container

image: nginx:latest

### **4. Service**

* **Description:** Exposes Pods to the network, enabling communication within and outside the cluster.
* **Types:** ClusterIP, NodePort, LoadBalancer.

**Example:** apiVersion: v1

kind: Service

metadata:

name: my-service

spec:

selector:

app: nginx

ports:

- protocol: TCP

port: 80

targetPort: 80

type: ClusterIP

### **5. ConfigMap**

* **Description:** Stores configuration data as key-value pairs for use in Pods.

**Example:** apiVersion: v1

kind: ConfigMap

metadata:

name: my-config

data:

key1: value1

key2: value2

### **6. Secret**

* **Description:** Similar to ConfigMap but designed for sensitive data like passwords or tokens.

**Example:** apiVersion: v1

kind: Secret

metadata:

name: my-secret

type: Opaque

data:

username: bXl1c2Vy # Base64 encoded

password: bXlwYXNzd29yZA==

### **7. PersistentVolume (PV) and PersistentVolumeClaim (PVC)**

* **Description:** Abstracts storage in Kubernetes.

**Example PV:** apiVersion: v1

kind: PersistentVolume

metadata:

name: my-pv

spec:

capacity:

storage: 1Gi

accessModes:

- ReadWriteOnce

hostPath:

path: /mnt/data

**Example PVC:** apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: my-pvc

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 1Gi

### **8. StatefulSet**

* **Description:** Manages stateful applications, ensuring ordered deployment and unique Pod identities.

**Example:** apiVersion: apps/v1

kind: StatefulSet

metadata:

name: my-statefulset

spec:

serviceName: "stateful-service"

replicas: 3

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:latest

### **9. DaemonSet**

* **Description:** Ensures one Pod runs on each node.

**Example:** apiVersion: apps/v1

kind: DaemonSet

metadata:

name: my-daemonset

spec:

selector:

matchLabels:

app: daemon

template:

metadata:

labels:

app: daemon

spec:

containers:

- name: my-daemon

image: my-daemon-image

### **10. Job and CronJob**

**Job Example:** apiVersion: batch/v1

kind: Job

metadata:

name: my-job

spec:

template:

spec:

containers:

- name: my-job-container

image: busybox

command: ["echo", "Hello, Kubernetes!"]

restartPolicy: Never

**CronJob Example:** apiVersion: batch/v1

kind: CronJob

metadata:

name: my-cronjob

spec:

schedule: "\*/5 \* \* \* \*"

jobTemplate:

spec:

template:

spec:

containers:

- name: cronjob-container

image: busybox

command: ["echo", "Hello from CronJob!"]

restartPolicy: Never

### **11. Ingress**

* **Description:** Manages HTTP/HTTPS access to services within a cluster.

**Example:** apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: my-ingress

spec:

rules:

- host: myapp.example.com

http:

paths:

- path: /

pathType: Prefix

backend:

service:

name: my-service

port:

number: 80

### **12. CustomResourceDefinition (CRD)**

* **Description:** Extends Kubernetes with custom objects.

**Example:** apiVersion: apiextensions.k8s.io/v1

kind: CustomResourceDefinition

metadata:

name: myresources.mygroup.example.com

spec:

group: mygroup.example.com

names:

kind: MyResource

listKind: MyResourceList

plural: myresources

singular: myresource

scope: Namespaced

versions:

- name: v1

served: true

storage: true

## **4. Summary**

This tutorial covers the major Kubernetes objects from basic to advanced. Each object plays a vital role in defining and managing your workloads. YAML files are used to declaratively define these objects, and proper indentation is critical for ensuring correct behavior. By understanding these objects, you can design robust Kubernetes architectures for a wide range of applications.