

What is Minikube?

Minikube is an **open-source tool** that allows running a **single-node Kubernetes cluster** locally on a machine. It is ideal for developers and learners who want to **experiment with Kubernetes** without setting up a complex multi-node cluster on the cloud.

Purpose of Minikube

- Minikube is mainly used for **learning Kubernetes concepts** and **testing applications** locally.
- It helps **developers deploy, manage, and debug** Kubernetes applications in a local environment.

Why Use Minikube?

1. Ease of Setup

Minikube **simplifies** Kubernetes installation by setting up a **lightweight virtual machine or container** that contains all required Kubernetes components.

2. Features of Minikube

- **Supports Kubernetes add-ons** (e.g., ingress, metrics-server, and dashboard).
- **Multi-cluster support** for testing multiple Kubernetes clusters at the same time.
- **Built-in Docker daemon**, eliminating the need for a separate Docker installation.
- **Configurable resource limits** (CPU and memory) to optimize performance.

3. Cross-Platform Compatibility

Minikube works on:

- **Windows**

- macOS
- Linux

4. Use Cases

-  Learning **Kubernetes basics** in a local environment.
-  Testing **CI/CD pipelines** and Kubernetes deployments.
-  Debugging **Kubernetes issues** locally before deploying to production.

5. Integration

Minikube integrates with the Kubernetes CLI tool **kubectl**, allowing users to interact with Kubernetes clusters easily.

Minikube Installation Guide

Step 1: Download Minikube

Run the following command to download Minikube for **Linux**:

```
bash
CopyEdit
curl -LO
https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64
```

Explanation:

- curl -LO: Downloads the latest Minikube binary from the official **GitHub** repository.

Step 2: Install Minikube

```
bash
```

CopyEdit

```
sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm  
minikube-linux-amd64
```

 **Explanation:**

- `sudo install minikube-linux-amd64 /usr/local/bin/minikube`: Installs Minikube globally.
- `rm minikube-linux-amd64`: Deletes the downloaded file after installation.

 **Common Error:**

- If you see `rm: cannot remove 'minikube-linux-'`: No such file or directory, it means the filename is **incorrectly split**.
 **Fix:** Remove the extra space in `minikube-linux- amd64` and use the correct filename `minikube-linux-amd64`.

Step 3: Start Minikube

bash

CopyEdit

```
minikube start
```

 **Explanation:**

- Starts the Minikube **Kubernetes cluster**.

Step 4: Check Minikube Status

bash

CopyEdit

```
minikube status
```

📌 **Expected Output:**

```
plaintext
CopyEdit
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

📌 **Explanation:**

- Ensures **Minikube is running properly.**

Step 5: Get Running Pods

```
bash
CopyEdit
kubectl get pod
```

📌 **Explanation:**

- Displays a list of **all running pods.**
- If no pods exist, the output will be:

```
plaintext
CopyEdit
No resources found in default namespace.
```

Step 6: Get Deployments

```
bash
CopyEdit
```

```
kubectl get deploy
```

✖ **Explanation:**

- Lists all **Kubernetes deployments** in the default namespace.
- If no deployments exist, the output will be:

```
plaintext  
CopyEdit  
No resources found in default namespace.
```

Step 7: Get ReplicaSets

```
bash  
CopyEdit  
kubectl get replicaset
```

✖ **Explanation:**

- Lists all **ReplicaSets** that manage pod replicas.
- If no ReplicaSets exist, the output will be:

```
plaintext  
CopyEdit  
No resources found in default namespace.
```

✗ **Common Error:**

```
plaintext  
CopyEdit  
error: the server doesn't have a resource type "replica"
```

✓ **Fix:** Use `kubectl get replicaset` or the short form `kubectl get rs`.

Step 8: Get Detailed Pod Information

```
bash
CopyEdit
kubectl get pod -o wide
```

Explanation:

- Shows **detailed information** about running pods, including:
 - Node name
 - IP address
 - Status
 - Restarts

```

arulmurugan-19@Arul:~ Verifying Kubernetes components...
  • Using image gcr.io/k8s-minikube/storage-provisioner:v5
  • Enabled addons: storage-provisioner, default-storageclass
  • Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
arulmurugan-19@Arul:~$ minikube status
minikube
type: Control Plane
host: Running
kublet: Running
apiserver: Running
kubeconfig: Configured

arulmurugan-19@Arul:~$ kubectl get pod
No resources found in default namespace.
arulmurugan-19@Arul:~$ sudo nano pod.yaml
arulmurugan-19@Arul:~$ kubectl get pod
No resources found in default namespace.
arulmurugan-19@Arul:~$ kubectl get pods
No resources found in default namespace.
arulmurugan-19@Arul:~$ kubectl apply -f pod.yaml
pod/my-pod created
arulmurugan-19@Arul:~$ kubectl get pod
NAME READY STATUS RESTARTS AGE
my-pod 0/1 InvalidImageName 0 20s
arulmurugan-19@Arul:~$ kubectl get deploy
No resources found in default namespace.
arulmurugan-19@Arul:~$ sudo nano deploy.yaml
arulmurugan-19@Arul:~$ kubectl get deploy
No resources found in default namespace.
arulmurugan-19@Arul:~$ kubectl apply -f deploy.yaml
deployment.apps/my-deploy created
service/my-service created
arulmurugan-19@Arul:~$ kubectl get deploy
NAME READY UP-TO-DATE AVAILABLE AGE
my-deploy 0/5 5 0 5s
arulmurugan-19@Arul:~$ sudo nano re-test.yaml
arulmurugan-19@Arul:~$ kubectl apply -f re-test.yaml
replicaset.apps/my-rs created
arulmurugan-19@Arul:~$ kubectl get re-test
error: the server doesn't have a resource type "re-test"
arulmurugan-19@Arul:~$ kubectl get replica
arulmurugan-19@Arul:~$ kubectl get replica
error: the server doesn't have a resource type "replica"
arulmurugan-19@Arul:~$ kubectl get replicaset
NAME DESIRED CURRENT READY AGE
my-deploy-75f9b65756 5 5 5 6m25s
my-rs 4 4 4 76s
arulmurugan-19@Arul:~$ kubectl get pods
NAME READY STATUS RESTARTS AGE
my-deploy-75f9b65756-7x4w2 1/1 Running 0 6m37s
my-deploy-75f9b65756-86krf 1/1 Running 0 6m37s
my-deploy-75f9b65756-gdxxp 1/1 Running 0 6m37s
my-deploy-75f9b65756-mzsv4 1/1 Running 0 6m37s
my-deploy-75f9b65756-zqj98 1/1 Running 0 6m37s
my-pod 0/1 InvalidImageName 0 8m36s
my-rs-c4mbz 1/1 Running 0 88s
my-rs-m2w96 1/1 Running 0 88s
my-rs-rngt6 1/1 Running 0 88s
my-rs-wbfs4 1/1 Running 0 88s
arulmurugan-19@Arul:~$ kubectl get pod -o wide
NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES
my-deploy-75f9b65756-7x4w2 1/1 Running 0 10.244.0.5 minikube <none> <none>
my-deploy-75f9b65756-86krf 1/1 Running 0 10.244.0.6 minikube <none> <none>
my-deploy-75f9b65756-gdxxp 1/1 Running 0 10.244.0.4 minikube <none> <none>
my-deploy-75f9b65756-mzsv4 1/1 Running 0 10.244.0.8 minikube <none> <none>
my-deploy-75f9b65756-zqj98 1/1 Running 0 10.244.0.7 minikube <none> <none>
my-pod 0/1 InvalidImageName 0 10.244.0.3 minikube <none> <none>
my-rs-c4mbz 1/1 Running 0 97s 10.244.0.12 minikube <none> <none>
my-rs-m2w96 1/1 Running 0 97s 10.244.0.9 minikube <none> <none>
my-rs-rngt6 1/1 Running 0 97s 10.244.0.11 minikube <none> <none>
my-rs-wbfs4 1/1 Running 0 97s 10.244.0.10 minikube <none> <none>
arulmurugan-19@Arul:~|

```

Pod.yaml

apiVersion: v1

kind: Pod

```
metadata:  
  name: my-pod  
  
labels:  
  app: my-web-app  
  type: backend  
  
spec:  
  containers:  
    - name: nginx-container  
      image: nginx //our image name  
  ports:  
    - containerPort: 80
```

```
Deploy.yaml  
  
apiVersion: apps/v1  
kind: Deployment  
  
metadata:  
  name: my-deploy  
  
labels:  
  name: my-deploy  
  
spec:  
  replicas: 5  
  selector:  
    matchLabels:  
      apptype: web-backend  
  strategy:  
    type: RollingUpdate
```

```
template:

metadata:

labels:
    apptype: web-backend

spec:

containers:
    - name: my-app
        image: 22csr019/devops_dockerhub_1
        ports:
            - containerPort: 9000
```

```
apiVersion: v1
kind: Service
metadata:
    name: my-service
labels:
    app: my-service
spec:
    type: NodePort
    ports:
        - port: 9000
          targetPort: 8080
          nodePort: 30002
    selector:
```

```
apptype: web-backend
```

```
Re-test.yaml
```

```
apiVersion: apps/v1 # Add this line
```

```
kind: ReplicaSet
```

```
metadata:
```

```
  name: my-rs
```

```
  labels:
```

```
    name: my-rs
```

```
spec:
```

```
  replicas: 4
```

```
  selector:
```

```
    matchLabels:
```

```
      apptype: web-backend
```

```
template:
```

```
  metadata:
```

```
    labels:
```

```
      apptype: web-backend
```

```
spec:
```

```
  containers:
```

```
    - name: my-app
```

```
      image: 22csr019/devops_dockerhub_1
```

```
      ports:
```

```
        - containerPort: 9090
```