Kubernetes (Minikube) Installation Guide

**Minikube** is local Kubernetes, focusing on making it easy to learn and develop for Kubernetes

All you need is Docker (or similarly compatible) container or a Virtual Machine environment, and Kubernetes is a single command away: minikube start

# What you’ll need

· 2 CPUs or more

· 2GB of free memory

· 20GB of free disk space

· Internet connection

· Container or virtual machine manager, such as: [Docker](https://minikube.sigs.k8s.io/docs/drivers/docker/), [Hyperkit](https://minikube.sigs.k8s.io/docs/drivers/hyperkit/), [Hyper- V](https://minikube.sigs.k8s.io/docs/drivers/hyperv/), [KVM](https://minikube.sigs.k8s.io/docs/drivers/kvm2/), [Parallels](https://minikube.sigs.k8s.io/docs/drivers/parallels/), [Podman](https://minikube.sigs.k8s.io/docs/drivers/podman/), [VirtualBox](https://minikube.sigs.k8s.io/docs/drivers/virtualbox/), or [VMware Fusion/Workstation](https://minikube.sigs.k8s.io/docs/drivers/vmware/)

# 1. Installation

To install the latest minikube **stable** release on **x86-64 Linux** using **Debian package**:

curl-LO<https://storage.googleapis.com/minikube/releases/latest/minikube_latest_amd64.deb>

sudo dpkg -i minikube\_latest\_amd64.deb

# 2. Start your cluster

# From a terminal with administrator access (but not logged in as root), run:

# minikube start

If minikube fails to start, see the [drivers page](https://minikube.sigs.k8s.io/docs/drivers/) for help setting up a compatible

container or virtual-machine manager.

# 3. Interact with your cluster

If you already have kubectl installed, you can now use it to access your shiny new cluster:

kubectl get po -A

Alternatively, minikube can download the appropriate version of kubectl and you should be able to use it like this:

minikube kubectl -- get po -A

**You can also make your life easier by adding the following to your shell config:**

alias kubectl="minikube kubectl --"

Initially, some services such as the storage-provisioner, may not yet be in a Running state. This is a normal condition during cluster bring-up, and will resolve itself momentarily. For additional insight into your cluster state, minikube bundles the Kubernetes Dashboard, allowing you to get easily acclimated to your new environment:

minikube dashboard

4. **Deploy applications**

**Create a sample deployment and expose it on port 8080:**

Kubectl create deployment hello-minikube-image=k8s.gcr.io/echoserver

Kubectl expose deployment hello-minikube\_type=NodePort –port=8080

**It may take a moment, but your deployment will soon show up when you run:**

**kubectl get services hello-minikube**

**The easiest way to access this service is to let minikube launch a web browser for you:**

**minikube service hello-minikube**

**Alternatively, use kubectl to forward the port:**

**kubectl port-forward service/hello-minikube 7080:8080**

**Tada! Your application is now available at http://localhost:7080/.**

**You should be able to see the request metadata from nginx such as the CLIENT VALUES, SERVER VALUES, HEADERS RECEIVED and the BODY in the application output. Try changing the path of the request and observe the changes in the CLIENT VALUES. Similarly, you can do a POST request to the same and observe the body show up in BODY section of the output.**

**5. LoadBalancer deployments**

**To access a LoadBalancer deployment, use the “minikube tunnel” command. Here is an example deployment:**

**Kubectlcreatedeploymentbalanced--image=k8s.gcr.io/echoserver:1.4**

**kubectl expose deployment balanced --type=LoadBalancer --port=8080**

**In another window, start the tunnel to create a routable IP for the ‘balanced’ Deployment**

**To find the routable IP, run this command and examine the EXTERNAL-IP column:**

**kubectl get services balanced**

**Your deployment is now available at <EXTERNAL-IP>:8080**