**<https://github.com/kubernetes/minikube>**

**Minikube**

[](https://github.com/kubernetes/minikube/raw/master/logo/logo.png)

**What is Minikube?**

Minikube is a tool that makes it easy to run Kubernetes locally. Minikube runs a single-node Kubernetes cluster inside a VM on your laptop for users looking to try out Kubernetes or develop with it day-to-day.

**Installation**

**macOS**

brew cask install minikube

**Linux**

curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64 && chmod +x minikube && sudo cp minikube /usr/local/bin/ && rm minikube

**Windows**

Hyper-V needs to be enabled. For Windows 10 this can only run on these versions:

* Windows 10 Enterprise
* Windows 10 Professional
* Windows 10 Education

**Install with**[**Chocolatey**](https://chocolatey.org/)**(recommended):**

These commands must be run as administrator. To do this, open the Windows command line by typing 'cmd' in your start menu, right clicking it and choosing 'Run as administrator'.

choco install minikube

choco install kubernetes-cli

After it finished installing, close the current command line and restart. Minikube was added to your path automatically.

To start the minikube cluster, make sure you also have administrator rights.

minikube start

You might have to specify the vm driver.

minikube start --vm-driver hyperv

**Install manually**

Download the [minikube-windows-amd64.exe](https://storage.googleapis.com/minikube/releases/latest/minikube-windows-amd64.exe) file, rename it to minikube.exe and add it to your path.

**Linux Continuous Integration without VM Support**

Example with kubectl installation:

curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64 && chmod +x minikube && sudo cp minikube /usr/local/bin/ && rm minikube

curl -Lo kubectl https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl && chmod +x kubectl && sudo cp kubectl /usr/local/bin/ && rm kubectl

export MINIKUBE\_WANTUPDATENOTIFICATION=false

export MINIKUBE\_WANTREPORTERRORPROMPT=false

export MINIKUBE\_HOME=$HOME

export CHANGE\_MINIKUBE\_NONE\_USER=true

mkdir -p $HOME/.kube

touch $HOME/.kube/config

export KUBECONFIG=$HOME/.kube/config

sudo -E minikube start --vm-driver=none

# this for loop waits until kubectl can access the api server that Minikube has created

for i in {1..150}; do # timeout for 5 minutes

kubectl get po &> /dev/null

if [ $? -ne 1 ]; then

break

fi

sleep 2

done

# kubectl commands are now able to interact with Minikube cluster

**Also:**

**Wget https://storage.googleapis.com/kubernetes-release/release/v1.4.3/bin/darwin/amd64/kubectl**

**Other Ways to Install**

* [Linux]
  + [Arch Linux AUR](https://aur.archlinux.org/packages/minikube/)
  + [Fedora/CentOS/Red Hat COPR](https://copr.fedorainfracloud.org/coprs/antonpatsev/minikube-rpm/)
  + [openSUSE/SUSE Linux Enterprise](https://build.opensuse.org/package/show/Virtualization:containers/minikube)
* [Windows] Download the [minikube-windows-amd64.exe](https://storage.googleapis.com/minikube/releases/latest/minikube-windows-amd64.exe) file, rename it to minikube.exe and add it to your path.

**Minikube Version Management**

The [asdf](https://github.com/asdf-vm/asdf) tool offers version management for a wide range of languages and tools. On macOS, [asdf](https://github.com/asdf-vm/asdf) is available via Homebrew and can be installed with brew install asdf. Then, the Minikube plugin itself can be installed with asdf plugin-add minikube. A specific version of Minikube can be installed with asdf install minikube <version>. The tool allows you to switch versions for projects using a .tool-versions file inside the project. An asdf plugin exists for kubectl as well.

We also released a Debian package and Windows installer on our [releases page](https://github.com/kubernetes/minikube/releases). If you maintain a Minikube package, please feel free to add it here.

**Requirements**

* [kubectl](https://kubernetes.io/docs/tasks/kubectl/install/)
* macOS
  + [Hyperkit driver](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#hyperkit-driver), [xhyve driver](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md" \l "xhyve-driver), [VirtualBox](https://www.virtualbox.org/wiki/Downloads), or [VMware Fusion](https://www.vmware.com/products/fusion)
* Linux
  + [VirtualBox](https://www.virtualbox.org/wiki/Downloads) or [KVM](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#kvm-driver)
  + **NOTE:** Minikube also supports a --vm-driver=none option that runs the Kubernetes components on the host and not in a VM. Docker is required to use this driver but no hypervisor. If you use --vm-driver=none, be sure to specify a [bridge network](https://docs.docker.com/network/bridge/#configure-the-default-bridge-network) for docker. Otherwise it might change between network restarts, causing loss of connectivity to your cluster.
* Windows
  + [VirtualBox](https://www.virtualbox.org/wiki/Downloads) or [Hyper-V](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#hyperV-driver)
* VT-x/AMD-v virtualization must be enabled in BIOS
* Internet connection on first run

**Quickstart**

Here's a brief demo of Minikube usage. If you want to change the VM driver add the appropriate --vm-driver=xxx flag to minikube start. Minikube supports the following drivers:

* virtualbox
* vmwarefusion
* [KVM2](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#kvm2-driver)
* [KVM (deprecated in favor of KVM2)](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#kvm-driver)
* [hyperkit](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#hyperkit-driver)
* [xhyve](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#xhyve-driver)
* [hyperv](https://github.com/kubernetes/minikube/blob/master/docs/drivers.md#hyperV-driver)
* none (**Linux-only**) - this driver can be used to run the Kubernetes cluster components on the host instead of in a VM. This can be useful for CI workloads which do not support nested virtualization.

$ minikube start

Starting local Kubernetes v1.7.5 cluster...

Starting VM...

SSH-ing files into VM...

Setting up certs...

Starting cluster components...

Connecting to cluster...

Setting up kubeconfig...

Kubectl is now configured to use the cluster.

$ kubectl run hello-minikube --image=k8s.gcr.io/echoserver:1.4 --port=8080

deployment "hello-minikube" created

$ kubectl expose deployment hello-minikube --type=NodePort

service "hello-minikube" exposed

# We have now launched an echoserver pod but we have to wait until the pod is up before curling/accessing it

# via the exposed service.

# To check whether the pod is up and running we can use the following:

$ kubectl get pod

NAME READY STATUS RESTARTS AGE

hello-minikube-3383150820-vctvh 1/1 ContainerCreating 0 3s

# We can see that the pod is still being created from the ContainerCreating status

$ kubectl get pod

NAME READY STATUS RESTARTS AGE

hello-minikube-3383150820-vctvh 1/1 Running 0 13s

# We can see that the pod is now Running and we will now be able to curl it:

$ curl $(minikube service hello-minikube --url)

CLIENT VALUES:

client\_address=192.168.99.1

command=GET

real path=/

...

$ kubectl delete service hello-minikube

service "hello-minikube" deleted

$ kubectl delete deployment hello-minikube

deployment "hello-minikube" deleted

$ minikube stop

Stopping local Kubernetes cluster...

Machine stopped.

**Interacting With Your Cluster**

**kubectl**

The minikube start command creates a "[kubectl context](https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands" \l "-em-set-context-em-)" called "minikube". This context contains the configuration to communicate with your Minikube cluster.

Minikube sets this context to default automatically, but if you need to switch back to it in the future, run:

kubectl config use-context minikube,

or pass the context on each command like this: kubectl get pods --context=minikube.

**Dashboard**

To access the [Kubernetes Dashboard](http://kubernetes.io/docs/user-guide/ui/), run this command in a shell after starting Minikube to get the address:

minikube dashboard

**Services**

To access a service exposed via a node port, run this command in a shell after starting Minikube to get the address:

minikube service [-n NAMESPACE] [--url] NAME

**Design**

Minikube uses [libmachine](https://github.com/docker/machine/tree/master/libmachine) for provisioning VMs, and [kubeadm](https://github.com/kubernetes/kubeadm) to provision a kubernetes cluster

For more information about Minikube, see the [proposal](https://github.com/kubernetes/community/blob/master/contributors/design-proposals/cluster-lifecycle/local-cluster-ux.md).

**Additional Links**

* [**Advanced Topics and Tutorials**](https://github.com/kubernetes/minikube/blob/master/docs/README.md)
* [**Contributing**](https://github.com/kubernetes/minikube/blob/master/CONTRIBUTING.md)
* [**Development Guide**](https://github.com/kubernetes/minikube/blob/master/docs/contributors/README.md)

**Community**

* [**#minikube on Kubernetes Slack**](https://kubernetes.slack.com/)
* [**kubernetes-users mailing list**](https://groups.google.com/forum/#!forum/kubernetes-users)(If you are posting to the list, please prefix your subject with "minikube: ")