

Creating an Interactive Environment - Using Minikube to Create a Cluster

<https://kubernetes.io/docs/tutorials/kubernetes-basics/create-cluster/cluster-intro/>

The screenshot shows the Kubernetes documentation page. On the left is a navigation sidebar with a search bar and a list of links: Home, Getting started, Concepts, Tasks, and Tutorials. Under Tutorials, there is a link for 'Create a Cluster' which is highlighted with a red checkmark. A red circle with the number '1' is drawn around the 'Create a Cluster' link. The main content area has a breadcrumb trail: 'Kubernetes Documentation / Tutorials / Learn Kubernetes Basics / Create a Cluster / Using Minikube to Create a Cluster'. Below the breadcrumb is a blue header with the title 'Using Minikube to Create a Cluster'. The 'Objectives' section lists three bullet points: 'Learn what a Kubernetes cluster is.', 'Learn what Minikube is.', and 'Start a Kubernetes cluster using an online terminal.' The 'Kubernetes Clusters' section begins with the text: 'Kubernetes coordinates a highly available cluster of computers that are connected to work as a single unit. The abstractions in Kubernetes allow you to deploy containerized applications to a cluster without tying them specifically to individual machines. To make use of this new model of deployment, applications need to be packaged in a way that decouples them from individual hosts: they need to be containerized. Containerized applications are more flexible and available than in past deployment models, where applications were installed directly onto specific machines as packages deeply integrated into the host. **Kubernetes automates the distribution and scheduling of application containers across a cluster in a more efficient way.** Kubernetes is an open-source platform and is production-

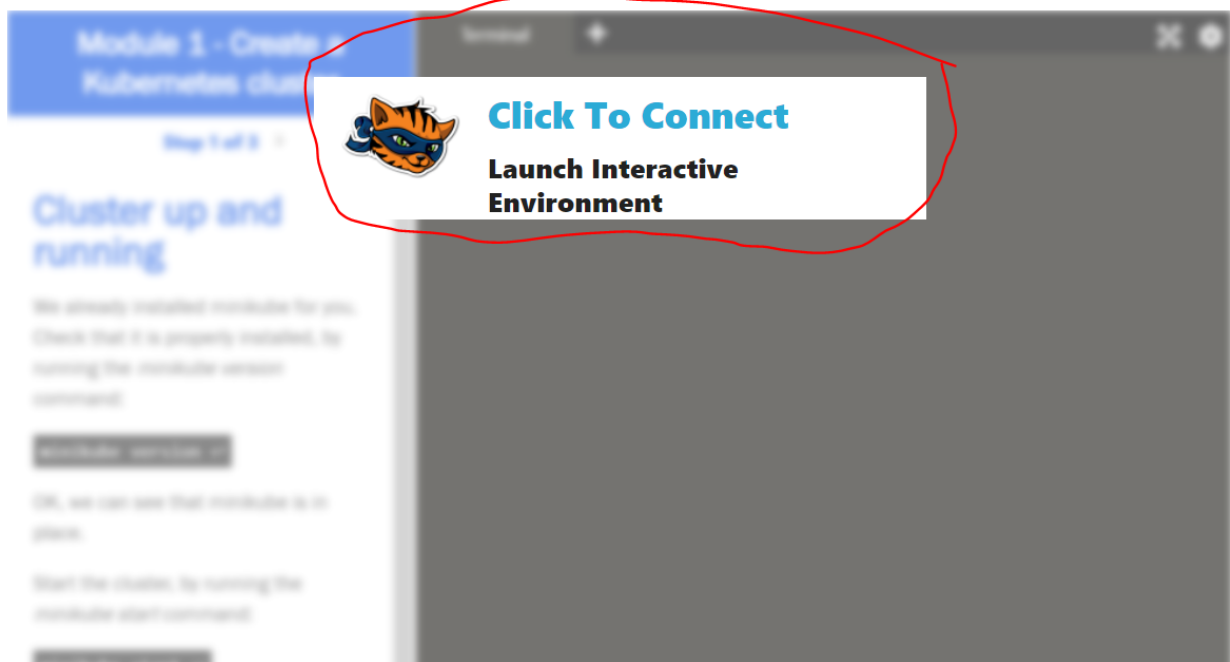
Scroll down and click on the Start Interactive Tutorial

Minikube is available for Linux, macOS, and Windows systems. The Minikube CLI provides basic bootstrapping operations for working with your cluster, including start, stop, status, and delete. For this tutorial, however, you'll use a provided online terminal with Minikube pre-installed.

Now that you know what Kubernetes is, let's go to the online tutorial and start our first cluster!

[Start Interactive Tutorial >](#)

Interactive Tutorial - Creating a Cluster



Click To Connect

Welcome!

Module 1 - Create a Kubernetes cluster

★ **Difficulty:** Beginner

🕒 **Estimated Time:** 10 minutes

The goal of this interactive scenario is to deploy a local development Kubernetes cluster using minikube

The online terminal is a pre-configured Linux environment that can be used as a regular console (you can type commands). Clicking on the blocks of code followed by the ENTER sign will execute that command in the terminal.

START SCENARIO

Click on Start Scenario

Module 1 - Create a Kubernetes cluster

Step 1 of 3 ▶

Cluster up and running

We already installed minikube for you. Check that it is properly installed, by running the *minikube version* command:

```
minikube version ↵
```

OK, we can see that minikube is in place.

Start the cluster, by running the *minikube start* command:

```
minikube start ↵
```

Great! You now have a running Kubernetes cluster in your online terminal. Minikube started a virtual

Terminal

\$

Next Click on Minikube Start

Module 1 - Create a Kubernetes cluster

Step 1 of 3 ►

Cluster up and running

We already installed minikube for you. Check that it is properly installed, by running the *minikube version* command:

```
minikube version ↵
```

OK, we can see that minikube is in place.

Start the cluster, by running the *minikube start* command:

```
minikube start ↵
```

Great! You now have a running Kubernetes cluster in your online terminal. Minikube started a virtual

Module 1 - Create a Kubernetes cluster

Step 1 of 3 ▶

Cluster up and running

We already installed minikube for you. Check that it is properly installed, by running the *minikube version* command:

```
minikube version ↵
```

OK, we can see that minikube is in place.

Start the cluster, by running the *minikube start* command:

```
minikube start ✓
```

Great! You now have a running Kubernetes cluster in your online terminal. Minikube started a virtual

```
Terminal +
$ minikube start
* minikube v1.18.0 on Ubuntu 18.04 (kvm/amd64)
* Using the none driver based on existing profile

X The requested memory allocation of 2200MiB does not leave room for
system overhead (total system memory: 2460MiB). You may face stability
issues.
* Suggestion: Start minikube with less memory allocated: 'minikube
start --memory=2200mb'

* Starting control plane node minikube in cluster minikube
* Running on localhost (CPUs=2, Memory=2460MB, Disk=194868MB) ...
* OS release is Ubuntu 18.04.5 LTS
* Preparing Kubernetes v1.20.2 on Docker 19.03.13 ...
- kubelet.resolv-conf=/run/systemd/resolve/resolv.conf
█
```

Wait for the minikube cluster to start

Module 1 - Create a Kubernetes cluster

Step 1 of 3 ▶

Cluster up and running

We already installed minikube for you. Check that it is properly installed, by running the *minikube version* command:

```
minikube version ↵
```

OK, we can see that minikube is in place.

Start the cluster, by running the *minikube start* command:

```
minikube start ✓
```

Great! You now have a running Kubernetes cluster in your online terminal. Minikube started a virtual

```
Terminal +
$ minikube start
* minikube v1.18.0 on Ubuntu 18.04 (kvm/amd64)
* Using the none driver based on existing profile

X The requested memory allocation of 2200MiB does not leave room for
system overhead (total system memory: 2460MiB). You may face stability
issues.
* Suggestion: Start minikube with less memory allocated: 'minikube
start --memory=2200mb'

* Starting control plane node minikube in cluster minikube
* Running on localhost (CPUs=2, Memory=2460MB, Disk=194868MB) ...
* OS release is Ubuntu 18.04.5 LTS
* Preparing Kubernetes v1.20.2 on Docker 19.03.13 ...
  - kubelet.resolv-conf=/run/systemd/resolve/resolv.conf
  - Generating certificates and keys ...
  - Booting up control plane ...
  - Configuring RBAC rules ...
* Configuring local host environment ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v4
* Enabled addons: default-storageclass, storage-provisioner
* Done! kubectl is now configured to use "minikube" cluster and "default"
namespace by default
$
```

The minikube is now ready

Module 1 - Create a Kubernetes cluster

Step 1 of 3 ▶

Cluster up and running

We already installed minikube for you. Check that it is properly installed, by running the *minikube version* command:

```
minikube version ↵
```

OK, we can see that minikube is in place.

Start the cluster, by running the *minikube start* command:

```
minikube start ✓
```

Great! You now have a running Kubernetes cluster in your online

```
Terminal +
$ kubectl get svc
NAME      TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
kubernetes ClusterIP  10.96.0.1      <none>         443/TCP     4m51s

$ kubectl get nodes
NAME      STATUS    ROLES    AGE   VERSION
minikube  Ready    control-plane,master  5m   v1.20.2

$ kubectl get pods
No resources found in default namespace.

$ kubectl get namespace
NAME              STATUS    AGE
default           Active    5m16s
kube-node-lease   Active    5m20s
kube-public       Active    5m20s
kube-system       Active    5m21s

$ kubectl cluster-info
Kubernetes control plane is running at https://10.0.0.14:8443
KubeDNS is running at https://10.0.0.14:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
$
```