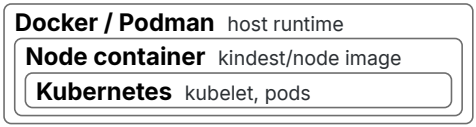
kind Kubernetes in Docker

CONCEPTS

Each kind node is a container. Your cluster lives inside Docker (or Podman), so it needs no VM and starts in under a minute.



Loaded images with a :latest tag are pulled anyway. Use a versioned tag or imagePullPolicy: IfNotPresent.

Contexts are named kind-<name>, so the default cluster is kind-kind.

CLUSTERS

kind create cluster
Create a cluster named kind

kind create cluster --name *<name>*
Create a named cluster

kind get clusters
List clusters

kind delete cluster
Delete the default cluster

kind delete cluster --name *<name>*
Delete a named cluster

kind create cluster --image kindest/node:*<tag>*
Pick the Kubernetes version

kind create cluster --wait 5m
Block until the control plane is ready

kind get nodes --name *<name>*
List node containers

kind version
Show the kind version

CONFIG FILE

kind create cluster --config *<file>*
Create from a config file

nodes: [{role: control-plane}, {role: worker}]
Add worker nodes

extraPortMappings: [{containerPort, hostPort}]
Publish a node port on the host

IMAGES

kind load docker-image *<image>*
Copy a local image into the nodes

kind load docker-image *<image>* **--name** *<name>*
Load into a named cluster

kind load image-archive *<file.tar>*
Load an image from a tarball

KUBECONFIG

kubectl cluster-info --context kind-*<name>*
Check you can reach the cluster

kubectl config use-context kind-*<name>*
Switch kubectl to this cluster

kind get kubeconfig --name *<name>*
Print the kubeconfig

kind export kubeconfig --name *<name>*
Write the context into your kubeconfig

DEBUGGING

kind export logs *<dir>*
Dump node and pod logs to a folder

docker exec -it *<name>***-control-plane bash**
Shell into a node