Kompose

Compose to Kubernetes

START HERE

`kompose convert`
Convert compose.yaml

`kompose convert -f <file>`
Pick the Compose file

`kompose convert -o <dir>`
Write to a directory

`kompose convert --stdout`
Print, write nothing

`kompose convert -n <ns>`
Set the namespace

`kompose convert --controller statefulset`
Pick the workload kind

`kubectl apply -f <dir>`
Deploy the output

`kompose convert --help`
All options

Compose services → `convert` → **Manifests** Deployment, Service, PVC → `apply` → **Cluster** running pods

CONCEPTS

Kompose turns a Compose file into Kubernetes manifests; it generates files and never talks to a cluster. Review the output, then apply it with kubectl.

Labels under a service's labels: key in the Compose file (kompose.service.type and friends) steer the output without extra flags.

Generated manifests are a starting point: check ports, volume sizes and probes before production.

CONVERT

`kompose convert`
Convert compose.yaml in this directory
Writes one YAML file per object into the current directory.

`kompose convert -f <file>`
Convert a specific Compose file

CONVERT (cont.)

`kompose -f <a> -f <b> convert`
Merge several files, later wins

`kompose convert -o <path>`
Save to a file or directory

`kompose convert --stdout`
Print objects instead of writing

`kompose convert --json`
Output JSON instead of YAML

`kompose convert -n <namespace>`
Set namespace on generated objects

`kompose convert --provider openshift`
Target OpenShift instead

`kompose convert --profile <name>`
Include services of a Compose profile

`kompose convert --indent <n>`
YAML indentation width

`kompose convert --no-interpolate`
Keep ${VAR} names as written

CONVERT (cont.)

`kompose convert --with-kompose-annotation=false`
Drop kompose.* annotations

WORKLOADS

`kompose convert --controller deployment`
Generate Deployments
Without a flag, restart: always or unset gives a controller object.

`kompose convert --controller daemonSet`
Generate DaemonSets

`kompose convert --replicas <n>`
Replica count in generated specs

`kompose convert --service-group-mode label`
Group services into one pod
Services share a pod by their kompose.service.group label.

`kompose convert --generate-network-policies`
Also emit NetworkPolicy objects

WORKLOADS (cont.)

Compose restart: to Kubernetes

restart	Object	Pod restartPolicy
`unset / always`	controller	Always
`unless-stopped`	controller	Always
`on-failure`	Pod or CronJob	OnFailure
`no`	Pod or CronJob	Never

VOLUMES & SECRETS

`kompose convert --volumes persistentVolumeClaim`
Volumes become PVCs (default)

`kompose convert --volumes emptyDir`
Use emptyDir volumes

`kompose convert --volumes hostPath`
Use hostPath volumes
Ties pods to one node; fine for local clusters only.

`kompose convert --pvc-request-size <size>`
Storage size of each PVC

VOLUMES & SECRETS (cont.)

`kompose convert --secrets-as-files`
Mount secrets as plain files

IMAGES

`kompose convert --build local`
Build images for build: services

`kompose convert --build local --push-image`
Build and push the images

`kompose convert --push-image-registry <registry>`
Push to another registry

HELM

`kompose convert -c`
Generate a Helm chart
The flag is hidden from --help in current source but still works; check your version.

`kompose convert -c -o <dir>`
Chart into a chosen directory

kompose.* labels

Label	Effect
`service.type`	nodeport, clusterip, loadbalancer, headless
`service.expose`	Make an Ingress (true or host)
`service.nodeport.port`	Fixed NodePort
`controller.type`	deployment, daemonset, ...
`volume.type`	PVC, emptyDir, hostPath, configMap
`volume.size`	PVC size, e.g. 1Gi
`image-pull-policy`	Always, IfNotPresent, Never
`service.group`	Share one pod

LABELS

More kompose.* labels

Label	Effect
`hpa.replicas.min/max`	Autoscaler bounds
`hpa.cpu`	CPU target, e.g. 50%
`cronjob.schedule`	Run as CronJob
`service.expose.tls-secret`	Ingress TLS secret
`image-pull-secret`	Registry secret

RECIPES

Convert and deploy

```
kompose convert -o k8s/
kubectl apply -f k8s/
kubectl get pods
```

HELP

`kompose --help`
Top-level help

`kompose version`
Print the version

`kompose completion bash`
Shell completion script