

# REFERÊNCIA RÁPIDA DE KUBERNETES

kubectl, pods, deployments, services, configs, depuração

## Básico do kubectl

### Informações do Cluster

```
kubectl cluster-info
kubectl get nodes
kubectl config current-context
kubectl config use-context my-cluster
```

### Comandos Essenciais

<b>kubectl get &lt;resource&gt;</b>	Listar recursos
<b>kubectl describe &lt;resource&gt; &lt;name&gt;</b>	Informações detalhadas do recurso
<b>kubectl create -f file.yaml</b>	Criar recurso a partir de arquivo
<b>kubectl apply -f file.yaml</b>	Criar ou atualizar recurso
<b>kubectl delete -f file.yaml</b>	Excluir recurso a partir de arquivo
<b>kubectl edit &lt;resource&gt; &lt;name&gt;</b>	Editar recurso no local
<b>kubectl api-resources</b>	Listar todos os tipos de recurso

### Formatos de Saída

<b>-o wide</b>	Colunas extras (IP, nó)
<b>-o yaml</b>	Saída YAML completa
<b>-o json</b>	Saída JSON completa
<b>-o jsonpath='{.spec}'</b>	Extrair campos específicos
<b>--sort-by=.metadata.name</b>	Ordenar saída por campo

## Pods

### Operações com Pods

```
kubectl get pods
kubectl get pods -A # all namespaces
kubectl run nginx --image=nginx # quick pod
kubectl delete pod nginx
```

### YAML de Pod

```
apiVersion: v1
kind: Pod
metadata:
  name: myapp
  labels: { app: myapp }
spec:
  containers:
    - name: app
      image: nginx:1.27
      ports:
        - containerPort: 80
```

### Valores de Status de Pod

<b>Running</b>	Todos os containers iniciados
<b>Pending</b>	Aguardando agendamento ou pull da imagem
<b>CrashLoopBackOff</b>	Container fica quebrando e reiniciando
<b>ImagePullBackOff</b>	Não é possível fazer pull da imagem do container
<b>Completed</b>	Executou até a conclusão (Jobs)

## Deployments

### YAML de Deployment

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web
spec:
  replicas: 3
  selector:
    matchLabels: { app: web }
  template:
    metadata:
      labels: { app: web }
    spec:
      containers:
        - name: web
          image: nginx:1.27
          ports:
            - containerPort: 80
```

### Comandos de Deployment

<b>kubectl get deploy</b>	Listar deployments
<b>kubectl scale deploy web --replicas=5</b>	Escalar réplicas
<b>kubectl set image deploy/web web=nginx:1.28</b>	Atualizar imagem (rolling)
<b>kubectl rollout status deploy/web</b>	Monitorar progresso do rollout
<b>kubectl rollout undo deploy/web</b>	Reverter para revisão anterior
<b>kubectl rollout history deploy/web</b>	Ver histórico de revisões

## Services

### Tipos de Service

<b>ClusterIP</b>	Somente interno (padrão)
<b>NodePort</b>	Expõe no IP de cada nó em porta estática
<b>LoadBalancer</b>	Load balancer externo (nuvem)
<b>ExternalName</b>	Aliás DNS para serviço externo

### YAML de Service

```
apiVersion: v1
kind: Service
metadata:
  name: web-svc
spec:
  type: ClusterIP
  selector: { app: web }
  ports:
    - port: 80
      targetPort: 80
```

### Expose Rápido

```
kubectl expose deploy web --port=80 --type=ClusterIP
kubectl expose deploy web --port=80 --type=NodePort
kubectl get svc
```

## ConfigMaps e Secrets

### ConfigMap

```
kubectl create configmap app-cfg \
--from-literal=DB_HOST=db.example.com \
--from-file=config.ini
```

### Secret

```
kubectl create secret generic db-creds \
--from-literal=username=admin \
--from-literal=password=s3cret
```

### Usando em Pods

```
# As environment variables
envFrom:
  - configMapRef: { name: app-cfg }
  - secretRef: { name: db-creds }

# As volume mount
volumes:
  - name: cfg
    configMap: { name: app-cfg }
```

### Comandos

<b>kubectl get cm</b>	Listar ConfigMaps
<b>kubectl get secret</b>	Listar Secrets
<b>kubectl describe cm app-cfg</b>	Exibir dados do ConfigMap
<b>kubectl get secret db-creds -o yaml</b>	Exibir Secret (codificado em base64)

## Namespaces

### Comandos de Namespace

<b>kubectl get ns</b>	Listar namespaces
<b>kubectl create ns staging</b>	Criar namespace
<b>kubectl delete ns staging</b>	Excluir namespace e todos os recursos
<b>kubectl get pods -n staging</b>	Listar pods no namespace
<b>kubectl get pods -A</b>	Listar pods em todos os namespaces

### Definir Namespace Padrão

```
kubectl config set-context --current \
--namespace=staging
```

## Volumes

### PersistentVolumeClaim

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: data-pvc
spec:
  accessModes: [ReadWriteOnce]
  resources:
    requests: { storage: 10Gi }
```

### Montar em Pod

```
volumes:
  - name: data
    persistentVolumeClaim:
      claimName: data-pvc
containers:
  - volumeMounts:
    - name: data
      mountPath: /app/data
```

### Tipos de Volume

<b>emptyDir</b>	Diretório temporário, excluído com o pod
<b>hostPath</b>	Monta caminho do sistema de arquivos do host
<b>persistentVolumeClaim</b>	Armazenamento persistente (PVC)
<b>configMap</b>	Monta ConfigMap como arquivos
<b>secret</b>	Monta Secret como arquivos

## Ingress

### YAML de Ingress

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: web-ingress
spec:
  rules:
    - host: app.example.com
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: web-svc
                port: { number: 80 }
```

### Notas sobre Ingress

<b>Ingress Controller</b>	Necessário (nginx-ingress, traefik, etc.)
<b>pathType: Prefix</b>	Corresponde ao prefixo da URL
<b>pathType: Exact</b>	Corresponde ao caminho exato da URL
<b>TLS</b>	Adicione seção `tls:` com nome do secret

## Depuração

### Comandos de Diagnóstico

<b>kubectl logs &lt;pod&gt;</b>	stdout/stderr do container
<b>kubectl logs &lt;pod&gt; -c &lt;ctr&gt;</b>	Logs de container específico
<b>kubectl logs &lt;pod&gt; --previous</b>	Logs de container que falhou
<b>kubectl describe pod &lt;pod&gt;</b>	Eventos, condições, status
<b>kubectl exec -it &lt;pod&gt; -- sh</b>	Shell no container
<b>kubectl port-forward &lt;pod&gt; 8080:80</b>	Encaminhar porta local para o pod
<b>kubectl top pods</b>	Uso de CPU/memória (metrics-server)
<b>kubectl get events --sort-by=.lastTimestamp</b>	Linha do tempo de eventos do cluster

### Pod de Depuração

```
kubectl run debug --rm -it --image=busybox -- sh
# or attach ephemeral container
kubectl debug -it <pod> --image=busybox
```

## Padrões Comuns

### Labels e Seletores

```
kubectl get pods -l app=web
kubectl get pods -l 'env in (prod,staging)'
```

### Limites de Recursos

```
resources:
  requests: { cpu: 100m, memory: 128Mi }
  limits: { cpu: 500m, memory: 256Mi }
```

### Liveness e Readiness

```
livenessProbe:
  httpGet: { path: /healthz, port: 8080 }
  initialDelaySeconds: 5
  periodSeconds: 10
readinessProbe:
  httpGet: { path: /ready, port: 8080 }
```

### Receitas Rápidas

<b>Dry run</b>	`kubectl apply -f file.yaml --dry-run=client`
<b>Gerar YAML</b>	`kubectl create deploy web --image=nginx --dry-run=client -o yaml`
<b>Watch</b>	`kubectl get pods -w`
<b>Copiar arquivos</b>	`kubectl cp file.txt pod:/tmp/`
<b>Reiniciar deploy</b>	`kubectl rollout restart deploy/web`