# Kubernetes Cheat Sheet

- **Cluster:** A group of connected computers (nodes) that run applications.
- **Node:** A single computer in a cluster that runs applications.
- **Pod:** The smallest unit in Kubernetes that can run one or more containers.
- **Namespace:** A way to divide resources in a cluster for different projects or teams.
- **Deployment:** Manages a set of identical pods to ensure the correct number are running.
- **ReplicaSet:** Ensures a specified number of pod copies are running at all times.
- **DaemonSet:** Ensures a pod runs on all or some nodes.
- **StatefulSet:** Manages stateful applications, keeping track of each pod's identity.
- **Job:** Runs a task until it completes successfully.
- **CronJob:** Runs tasks on a scheduled basis, like a cron job in Unix.
- **Service:** Exposes a set of pods as a network service.
- **Ingress:** Manages external access to services, usually HTTP.
- **ConfigMap:** Stores configuration data as key-value pairs.
- **Secret:** Stores sensitive data, like passwords and tokens.
- **Volume:** Provides storage for containers.
- **PersistentVolume (PV):** A piece of storage that an administrator sets up.
- **PersistentVolumeClaim (PVC):** A request for storage by a user.
- **Kubelet:** The agent that runs on each node to manage pods.
- **Kube-Proxy:** Manages network rules on nodes.
- **Controller Manager:** Manages controllers that regulate the state of the cluster.
- **Scheduler:** Decides which nodes will run new pods.
- **Etcd:** A key-value store that stores all cluster data.
- **Kubectl:** The command-line tool to interact with the Kubernetes API.
- **Helm:** A package manager for Kubernetes applications.
- **Horizontal Pod Autoscaler:** Automatically adjusts the number of pods based on resource usage.

- **Cluster Autoscaler:** Automatically adjusts the number of nodes in a cluster based on resource usage.
- **Label:** Key-value pairs attached to objects for organizing and selecting them.
- **Annotation:** Metadata attached to objects to provide additional information.
- **Taints:** Prevents specific pods from running on certain nodes.
- **Tolerations:** Allows pods to run on nodes with specific taints.
- **Affinity/Anti-Affinity:** Rules that specify which nodes can or cannot run specific pods.
- **Role-Based Access Control (RBAC):** Manages who can do what in the cluster.
- **ServiceAccount:** An identity for processes running in pods to interact with the Kubernetes API.
- **ClusterRole:** Defines permissions that apply across the entire cluster.
- **Role:** Defines permissions within a specific namespace.
- **RoleBinding:** Grants a Role's permissions to a user or group within a namespace.
- **ClusterRoleBinding:** Grants a ClusterRole's permissions to a user or group across the entire cluster.
- **NetworkPolicy:** Controls the traffic between pods in the cluster.
- **PodSecurityPolicy:** Defines security rules that pods must follow.
- **PodDisruptionBudget (PDB):** Limits the number of pods that can be unavailable during maintenance.
- **Ingress Controller:** Manages Ingress resources to provide HTTP and HTTPS routing.
- **CoreDNS:** A DNS server for the cluster, providing name resolution for services.
- **StorageClass:** Describes different types of storage available in the cluster.
- **Init Containers:** Special containers that run before the main containers in a pod start.
- **Sidecar Container:** A helper container that runs alongside the main container in a pod.
- **Readiness Probe:** Checks if a container is ready to start accepting traffic.
- **Liveness Probe:** Checks if a container is still running and should be restarted if not.
- **Headless Service:** A service without a cluster IP, used to directly access pods.
- **LoadBalancer Service:** Exposes a service externally using a cloud provider's load balancer.
- **ClusterIP Service:** Exposes a service internally within the cluster.

# Kubernetes
## Cheat Sheet (Part 2)

- **NodePort Service:** Exposes a service on a static port on each node.
- **Endpoints:** A list of IP addresses and ports that a service forwards traffic to.
- **Resource Quotas:** Limits the amount of resources a namespace can use.
- **LimitRange:** Defines resource usage limits for containers in a namespace.
- **Custom Resource Definition (CRD):** Extends Kubernetes to manage custom resources.
- **Operator:** A custom controller that manages complex applications on Kubernetes.
- **Admission Controller:** Intercepts requests to the Kubernetes API for validation and mutation.
- **Finalizer:** Ensures that specific cleanup steps are completed before an object is deleted.
- **Horizontal Pod Autoscaler (HPA):** Automatically scales the number of pods based on CPU/memory usage.
- **Vertical Pod Autoscaler (VPA):** Adjusts the resource limits and requests for running pods.
- **Cluster Autoscaler:** Automatically adjusts the size of the Kubernetes cluster by adding or removing nodes.
- **Affinity Rules:** Specify rules about which nodes can host a pod.
- **Anti-Affinity Rules:** Specify rules about which nodes should not host a pod.
- **Init Containers:** Special containers that run before the main containers in a pod start.
- **Sidecar Containers:** Helper containers that run alongside the main container in a pod.
- **Resource Requests:** Specify the minimum amount of resources a container needs.
- **Resource Limits:** Specify the maximum amount of resources a container can use.
- **PersistentVolumeClaim (PVC):** A request for storage by a user.
- **EmptyDir:** A temporary directory that is created when a pod is assigned to a node.
- **ConfigMap:** Provides configuration data to pods.
- **Secret:** Stores sensitive data, such as passwords and keys.
- **Security Context:** Defines security settings for a pod or container.
- **ServiceAccount:** Provides an identity for processes running in pods.
- **ClusterRoleBinding:** Binds a ClusterRole to a user or group for the entire cluster.
- **RoleBinding:** Binds a Role to a user or group within a namespace.
- **Pod Preset:** Injects certain information, like secrets or volume mounts, into pods at creation.
- **Priority Class:** Specifies the priority of pods to influence their scheduling.
- **Horizontal Pod Autoscaler:** Scales the number of pods based on observed CPU/memory utilization.
- **Vertical Pod Autoscaler:** Adjusts the CPU and memory requests/limits for pods.
- **Cluster Autoscaler:** Automatically adds/removes nodes based on cluster usage.
- **Self-healing:** Automatically replaces and reschedules failed containers.
- **Secrets Management:** Manages sensitive information like passwords and API keys.
- **Resource Quotas:** Limits the amount of resources that can be consumed in a namespace.
- **Default Namespace:** The default namespace for Kubernetes objects without a specified namespace.
- **Master Node:** Controls and manages the Kubernetes cluster.
- **Worker Node:** Runs applications and workloads in pods.
- **Controller Manager:** Runs controllers to regulate the state of the cluster.
- **Scheduler:** Assigns pods to nodes based on resource availability.
- **Etcd:** Stores all cluster data, ensuring data consistency and availability.
- **Kubelet:** Manages pod operations on each node.
- **Kube-Proxy:** Manages network rules and traffic routing for services.
- **Kubectl:** Command-line tool to interact with the Kubernetes API.
- **Helm:** Package manager for managing Kubernetes applications.
- **Helm Chart:** Pre-configured Kubernetes resources packaged for easy deployment.
- **Kustomize:** Tool for customizing Kubernetes YAML configurations.
- **Admission Controller:** Intercepts requests to the Kubernetes API for validation and mutation.
- **Custom Resource Definition (CRD):** Extends Kubernetes by defining custom resources.
- **Operator:** Custom controllers for managing complex applications.
- **Kubeadm:** Tool for initializing and managing Kubernetes clusters.
- **Minikube:** Tool for running a single-node Kubernetes cluster locally for testing and development.

# Most Useful kubectl Commands for Working with Kubernetes (Part 1)

| Command | Description |
|---|---|
| kubectl cluster-info | Provides information about the Kubernetes cluster. |
| kubectl get all -n <namespace> | Lists all resources in a specified namespace. |
| kubectl get nodes | Lists all nodes in the cluster. |
| kubectl get pods | Lists all pods in the default namespace. |
| kubectl get pods -n <namespace> | Lists all pods in the specified namespace. |
| kubectl describe pod <pod-name> | Shows detailed information about a specific pod. |
| kubectl get services | Lists all services in the default namespace. |
| kubectl get deployments | Lists all deployments in the default namespace. |
| kubectl describe deployment <deployment-name> | Shows detailed information about a specific deployment. |
| kubectl get replicasets | Lists all ReplicaSets in the default namespace. |
| kubectl get configmaps | Lists all ConfigMaps in the default namespace. |
| kubectl get secrets | Lists all secrets in the default namespace. |
| kubectl get namespaces | Lists all namespaces in the cluster. |
| kubectl create namespace <namespace-name> | Creates a new namespace. |
| kubectl delete namespace <namespace-name> | Deletes a namespace. |
| kubectl get pvc | Lists all PersistentVolumeClaims in the default namespace. |
| kubectl get pv | Lists all PersistentVolumes. |
| kubectl apply -f <file.yaml> | Applies a configuration file to create/update resources. |
| kubectl delete <resource-type> <resource-name> | Deletes a specific resource (e.g., pod, service). |
| kubectl scale deployment <deployment-name> --replicas=<number> | Scales a deployment to the specified number of replicas. |
| kubectl edit <resource-type> <resource-name> | Edits a resource configuration in real-time. |
| kubectl port-forward pod/<pod-name> <local-port>:<pod-port> | Forwards a port from a pod to your local machine. |
| kubectl exec -it <pod-name> -- /bin/bash | Executes a command in a running pod (e.g., opens a bash shell). |
| kubectl get events | Lists events in the cluster. |
| kubectl logs <pod-name> | Retrieves logs from a specific pod. |
| kubectl get <resource-type> -w | Watches for changes to resources (e.g., pods, services). |
| kubectl top pod | Displays resource usage for pods. |
| kubectl label <resource-type> <resource-name> <label-key>=<label-value> | Adds a label to a resource. |
| kubectl annotate <resource-type> <resource-name> <annotation-key>=<annotation-value> | Adds an annotation to a resource. |
| kubectl patch <resource-type> <resource-name> --patch '<patch-data>' | Updates a resource with the provided patch data. |
| kubectl rollout status deployment/<deployment-name> | Shows the rollout status of a deployment. |
| kubectl rollout undo deployment/<deployment-name> | Rolls back to a previous deployment version. |

# Most Useful kubectl Commands for Working with Kubernetes (Part 2)

| Command | Description |
|---|---|
| kubectl expose pod <pod-name> --port=<port> --target-port=<target-port> | Exposes a pod to the network as a new service. |
| kubectl get serviceaccounts | Lists all service accounts. |
| kubectl create secret generic <secret-name> --from-literal=<key>=<value> | Creates a secret from literal values. |
| kubectl create secret generic <secret-name> --from-file=<filename> | Creates a secret from a file. |
| kubectl create configmap <configmap-name> --from-file=<filename> | Creates a ConfigMap from a file. |
| kubectl api-resources | Lists available API resources. |
| kubectl api-versions | Lists available API versions. |
| kubectl patch <resource-type> <resource-name> --type <patch-type> --patch '<patch-data>' | Applies a patch to a resource. |
| kubectl delete pod -l <label-key>=<label-value> | Deletes pods based on a label selector. |
| kubectl cordon <node-name> | Marks a node as unschedulable. |
| kubectl uncordon <node-name> | Marks a node as schedulable. |
| kubectl drain <node-name> | Safely evicts all pods from a node. |
| kubectl create serviceaccount <service-account-name> | Creates a new service account. |
| kubectl get roles | Lists all roles in the default namespace. |
| kubectl get rolebindings | Lists all role bindings in the default namespace. |
| kubectl get clusterroles | Lists all cluster roles. |
| kubectl get clusterrolebindings | Lists all cluster role bindings. |
| kubectl create rolebinding <rolebinding-name> --role=<role-name> --serviceaccount=<namespace>:<service-account-name> -n <namespace> | Creates a role binding in a specific namespace. |
| kubectl apply -k <directory> | Applies a directory of configuration files. |
| kubectl create -f <file.yaml> | Creates resources defined in a configuration file. |
| kubectl replace -f <file.yaml> | Replaces a resource defined in a configuration file. |
| kubectl rollout restart deployment <deployment-name> | Restarts the pods in a deployment. |
| kubectl logs -f <pod-name> | Streams the logs from a specific pod. |
| kubectl cp <source> <destination> | Copies files and directories to and from containers. |
| kubectl get rs | Lists all ReplicaSets. |
| kubectl get hpa | Lists all Horizontal Pod Autoscalers. |
| kubectl get cronjobs | Lists all CronJobs. |
| kubectl get daemonsets | Lists all DaemonSets. |
| kubectl get statefulsets | Lists all StatefulSets. |
| kubectl delete pod --grace-period=<seconds> | Deletes a pod with a specified grace period. |
| kubectl edit deployment <deployment-name> | Edits an existing deployment. |

# Most Useful kubectl Commands for Working with Kubernetes (Part 3)

| Command | Description |
|---|---|
| kubectl rollout history deployment <deployment-name> | Shows the rollout history of a deployment. |
| kubectl get endpoints | Lists all endpoints in the default namespace. |
| kubectl get ingress | Lists all Ingress resources in the default namespace. |
| kubectl delete job <job-name> | Deletes a specific Job. |
| kubectl delete cronjob <cronjob-name> | Deletes a specific CronJob. |
| kubectl get resourcequotas | Lists all resource quotas in the default namespace. |
| kubectl get limistranges | Lists all limit ranges in the default namespace. |
| kubectl get csr | Lists all CertificateSigningRequests. |
| kubectl certificate approve <csr-name> | Approves a certificate signing request. |
| kubectl certificate deny <csr-name> | Denies a certificate signing request. |
| kubectl top node | Displays resource usage for nodes. |
| kubectl get events --all-namespaces | Lists all events in all namespaces. |
| kubectl port-forward service/<service-name> <local-port>:<service-port> | Forwards a port from a service to your local machine. |
| kubectl auth can-i <verb> <resource> | Checks if a user can perform an action. |
| kubectl drain <node-name> --ignore-daemonsets | Drains a node, ignoring DaemonSet-managed pods. |
| kubectl delete ingress <ingress-name> | Deletes a specific Ingress resource. |
| kubectl get priorityclasses | Lists all PriorityClasses. |
| kubectl create role <role-name> --verb=<verb> --resource=<resource> -n <namespace> | Creates a new role in a specific namespace. |
| kubectl create clusterrole <clusterrole-name> --verb=<verb> --resource=<resource> | Creates a new cluster role. |
| kubectl delete clusterrole <clusterrole-name> | Deletes a specific cluster role. |
| kubectl get csr <csr-name> -o jsonpath='{.status.conditions[?(@.type=="Approved")].status}' | Checks the approval status of a certificate signing request. |
| kubectl describe service <service-name> | Shows detailed information about a specific service. |
| kubectl describe ingress <ingress-name> | Shows detailed information about a specific Ingress resource. |
| kubectl create configmap <configmap-name> --from-env-file=<env-file> | Creates a ConfigMap from an environment file. |
| kubectl delete pv <pv-name> | Deletes a specific PersistentVolume. |
| kubectl delete pvc <pvc-name> | Deletes a specific PersistentVolumeClaim. |
| kubectl get horizontalpodautoscaler | Lists all Horizontal Pod Autoscalers in the default namespace. |