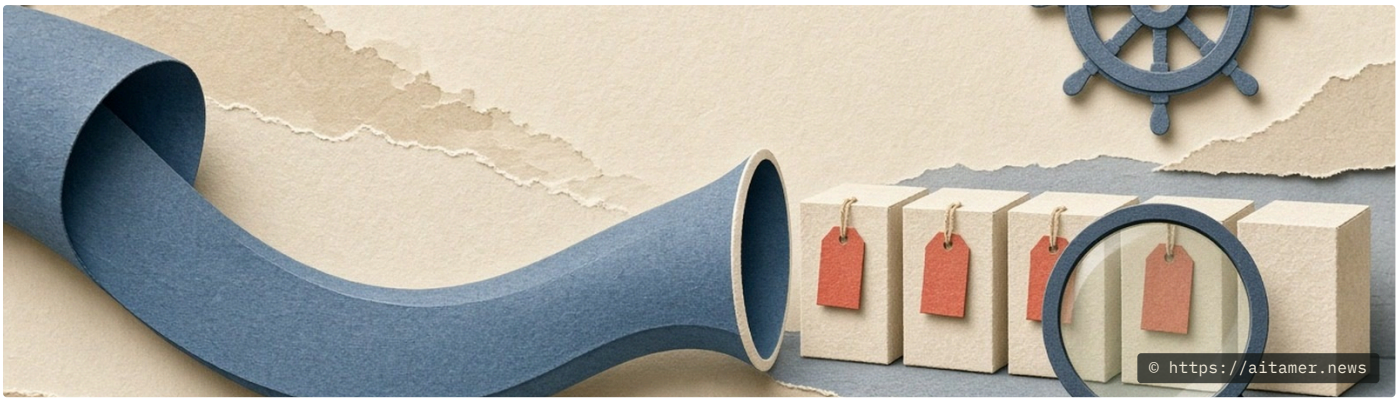


KUBERNETES FROM ZERO · FOUNDATIONS 5 OF 12 · LESSON 5 · LESSON KIT · ANSWER KEY

Kubernetes from Zero, part 5: kubectl essentials

The answers to the homework sheet, matched by exercise number. Every answer comes from the lesson. Names with random suffixes will differ on your cluster.

Every command ran on a kind v0.33.0 cluster (node v1.37.0, kubectl v1.37.2) on 10 Nov 2026. Cluster: k8s-ep04.

Recall

1. Name the three object-management techniques, and say which one the rest of the series uses.

   Imperative commands (flags on live objects), imperative object configuration (a file plus a verb you choose: `create -f`, `replace -f`, `delete -f`), and declarative object configuration (`kubectl apply -f` on files or directories). The series uses declarative configuration, paired with `kubectl diff`.

2. What rule does the object management page add about mixing techniques?

   Manage each object with only one technique. Mixing them on the same object gives results the docs call undefined.

3. What does kubectl itself store, and what does `-v=6` add?

   kubectl stores nothing. Every command becomes one or more HTTPS requests to the kube-apiserver. `-v=6` raises the log level enough to print each request. The lesson's line shows verb GET, the URL under `/api/v1/namespaces/k8s-from-zero/pods`, and status `200 OK`.

4. What do `--dry-run=client` and `--dry-run=server` each do?

   `--dry-run=client` builds the object locally and sends nothing. `--dry-run=server` sends the request through the API server's full validation and admission checks but stores nothing.

5. What three exit codes does `kubectl diff` use?

   `0` means no differences, `1` means differences, and more than `1` means an error. The lesson checked `0` before the edit and `1` after it.

6. In a label selector, what does a comma mean, and which objects does `!=` match besides the ones with a different value?

   A comma means AND, and there is no OR operator. Inequality also matches objects that do not have the key. In the lab, `tier!=frontend` matched scratch, which has no `tier` label.

Run this and explain

7. Run `kubectl run quick --image=nginx:1.29-alpine -n k8s-from-zero`. What does the lesson print, and which technique is this?

   `pod/quick created`. It is an imperative command: one line, no file left behind.

Free to share with credit: © aitamer.news

8 Run the `kubectl run web ... --dry-run=client -o yaml` command from step 2. Which labels does the lesson's YAML contain, and which two fields does the lesson say you can delete?

> `app.kubernetes.io/name: web` and `tier: frontend`. You can delete `resources: {}` and `status: {}`. The image is `nginx:1.29-alpine` and `containerPort` is 80.

9 Apply `web.yaml` twice. What does each run print?

> `pod/web created`, then `pod/web unchanged`. apply is idempotent.

10 Add `environment: dev` under `labels` and run `kubectl diff -f web.yaml`. What line does the lesson's diff add, and what is the exit code?

> The added line is `environment: dev`. The exit code is `1`, which means differences. apply then prints `pod/web configured`.

11 Run `kubectl explain pod.spec.containers.imagePullPolicy`. Which three values does the lesson list, and when is the default `Always`?

> `Always`, `IfNotPresent` and `Never`. The description in the lesson says the default is Always if the `:latest` tag is specified, or IfNotPresent otherwise. The field cannot be updated.

12 Apply `shop.yaml` and run the three equality selectors from the lesson (`tier=frontend`, `tier=frontend,environment=prod`, `tier!=frontend`). Which Pod names come back for each?

> `tier=frontend` returns shop-web-dev and shop-web-prod. Adding `environment=prod` returns only shop-web-prod. `tier!=frontend` returns scratch and shop-api-prod.

13 Run `kubectl label pod scratch -n k8s-from-zero environment=dev`, then the same command with `environment=test` and no `--overwrite`. What error does the lesson print?

> The first command prints `pod/scratch labeled`. The second prints `error: 'environment' already has a value (dev), and --overwrite is false`.

Predict the outcome

14 You run `kubectl create -f web.yaml` for a Pod named `web` that already exists. What error does the lesson quote?

> `Error from server (AlreadyExists): error when creating "web.yaml": pods "web" already exists`. `create` only creates. Use `apply` for files you keep changing.

15 You `kubectl edit` Pod web so `environment` is `test`, then `kubectl apply -f web.yaml` while the file still says `dev`. What does ENVIRONMENT show afterwards, and what happens to a label `owner=me` that the file never mentions?

> apply prints `pod/web configured` and ENVIRONMENT is `dev` again, because the file owns that field. `owner=me` stays. The lesson's `--show-labels` line lists `app.kubernetes.io/name=web,environment=dev,owner=me,tier=frontend`.

16 You JSON-patch Pod web so container `web` has `containerPort` 8080. What does the API do?

> It refuses the update. The lesson quotes: `The Pod "web" is invalid: spec: Forbidden: pod updates may not change fields other than` the image fields, `spec.activeDeadlineSeconds`, additions to `spec.tolerations`, and `spec.terminationGracePeriodSeconds` in the case the lesson names. For anything else you replace the Pod.

17 You run `kubectl get pods -n k8s-from-zero -l tier=Frontend`. What does the lesson say comes back, and why?

> Nothing. Values are case-sensitive, and the lesson says `-l tier=Frontend` found nothing. Check with `--show-labels` or `-L tier` first.

Free to share with credit: © aitamer.news

18 You are about to run `kubectl delete pods -n k8s-from-zero -l environment=dev`. What does the lesson tell you to run first, and which Pod did that selector delete in the lab?

> Run `kubectl get pods -l ...` with the same selector first. In the lab the delete removed `shop-web-dev`. A later `kubectl delete -f shop.yaml` reports `NotFound` for that Pod and removes the rest.

Free to share with credit: © aitamer.news