Kustomize Template-free manifests

Base plain manifests → `resources` → **Overlay** patches, images → `build` → **Rendered YAML** stdout → `apply -k` → **Cluster**

CONCEPTS

Kustomize never edits your YAML: it reads a base, applies declared transformations and prints the result.

Layout convention: base/ holds shared manifests, overlays/<env>/ each hold a kustomization.yaml that lists ../../base under resources.

BUILD AND APPLY

`kustomize build <dir>`
Render a kustomization to stdout

`kustomize build <dir> -o <path>`
Write output to a file or directory

`kubectl apply -k <dir>`
Build and apply to the cluster

`kubectl kustomize <dir>`
Build using kubectl's bundled kustomize

`kubectl diff -k <dir>`
Show what apply would change

`kubectl delete -k <dir>`
Delete everything the build produces

BUILD AND APPLY (cont.)

`kustomize build --enable-helm <dir>`
Inflate helmCharts entries

`kustomize version`
Print the version

CREATE AND EDIT

`kustomize create`
Write an empty kustomization.yaml

`kustomize create --autodetect`
Create it listing YAML files found

`kustomize edit add resource <path>`
Add a file or directory to resources

`kustomize edit add patch --path <file> --kind <kind>`
Add a patch entry

`kustomize edit add label <k>:<v>`
Add a label to all resources

`kustomize edit set image <name>=<newname>:<tag>`
Change an image name and tag

`kustomize edit set namespace <ns>`
Set the namespace

KUSTOMIZATION.YAML

Skeleton

```
apiVersion: kustomize.
config.k8s.io/v1beta1
kind: Kustomization
resources:
- ../../base
- extra.yaml
```

`resources:`
Files, directories or URLs to include

`namespace: <ns>`
Set namespace on every resource

`labels:`
Add labels, optionally to selectors

`replicas:`
Override replica counts

PATCHES

Strategic merge from a file

```
patches:
- path: patch.yaml
  target:
    kind: Deployment
    name: web
```

Inline JSON6902

```
patches:
- target: {kind: Deployment,
name: web}
  patch: |-
    - op: replace
      path: /spec/replicas
      value: 3
```

PATCHES (cont.)

`target: {kind, name, labelSelector}`
Select what a patch applies to

`op: add / replace / remove`
JSON6902 operations

Without a target, a strategic merge patch matches by the apiVersion, kind and name written in the patch itself. JSON6902 always needs a target.

IMAGES AND GENERATORS

Images

```
images:
- name: nginx
  newName: registry.example/
nginx
  newTag: "1.27"
```

`digest: sha256:<hash>`
Pin an image by digest instead of tag

ConfigMap from literals and files

```
configMapGenerator:
- name: app-config
  literals:
  - LOG_LEVEL=debug
  files:
  - app.properties
```

IMAGES AND GENERATORS (cont.)

`envs: [<file>.env]`
Load KEY=value lines as entries

`secretGenerator:`
Same fields as configMapGenerator, makes Secrets

`behavior: create / replace / merge`
How an overlay treats a base's generator

`options: {disableNameSuffixHash: true}`
Keep the plain name without a hash

REPLACEMENTS AND COMPONENTS

Copy a value into other fields

```
replacements:
- source:
    kind: ConfigMap
    name: app-config
    fieldPath: data.HOST
  targets:
  - select:
      kind: Deployment
    fieldPaths:
    - spec.template.spec.
containers.[name=web].env.
[name=HOST].value
```