kubeconform — Manifest schema validator

CONCEPTS

kubeconform checks manifests offline against JSON schemas; it never talks to a cluster, so admission rules and server-side defaults are not tested.

Manifests files, dirs, stdin
↓ parse
Schema lookup kind + apiVersion
↓ validate
Result valid / invalid / skipped

A kind with no schema is an error by default; use -ignore-missing-schemas or -skip, or add a CRD schema location.

RUNNING

`kubeconform` *<file>*
Validate one manifest

`kubeconform` *<dir>* *<file>*
Validate several paths at once

`kubeconform -`
Read manifests from stdin

`kubeconform -summary` *<dir>*
Print totals at the end

`kubeconform -verbose` *<dir>*
Show valid resources too

`kubeconform -exit-on-error` *<dir>*
Stop at the first failure

`kubeconform -ignore-filename-pattern '`*<regex>*`'` *<dir>*
Skip matching paths

`kubeconform -n 16` *<dir>*
Use more parallel workers

`kubeconform -v`
Print the version

`kubeconform -h`
Show all flags

STRICTNESS AND VERSIONS

`kubeconform -strict` *<dir>*
Fail on unknown or duplicate keys

`kubeconform -kubernetes-version 1.30.0` *<dir>*
Validate against that Kubernetes version

STRICTNESS AND VERSIONS (cont.)

`kubeconform -ignore-missing-schemas` *<dir>*
Skip kinds without a schema

`kubeconform -skip` *<kind>*`,`*<kind>* *<dir>*
Ignore kinds, all API versions

`kubeconform -skip v1/`*<Kind>* *<dir>*
Ignore one group/version of a kind

SCHEMAS AND CRDS

`kubeconform -schema-location default` *<dir>*
Use the built-in schema source

`kubeconform -schema-location '`*<url-or-path-template>*`'` *<dir>*
Add a custom schema location

`kubeconform -cache` *<dir>* *<manifests>*
Cache downloaded schemas on disk

Template variables for -schema-location

Variable	Expands to
`{{.ResourceKind}}`	Kind of the resource
`{{.ResourceAPIVersion}}`	Version part of apiVersion
`{{.Group}}`	API group
`{{.NormalizedKubernetesVersion}}`	Version with a v prefix
`{{.StrictSuffix}}`	-strict when -strict is set
`{{.KindSuffix}}`	Group and version suffix

OUTPUT

`kubeconform -output pretty` *<dir>*
Coloured, human-readable output

`kubeconform -output json` *<dir>*
Machine-readable result