Argo Workflows

Kubernetes workflow engine

START HERE

`argo lint <file>`
Validate a manifest

`argo submit <file> --watch`
Run and follow it

`argo list`
List workflows

`argo get @latest`
Show the newest run

`argo logs @latest -f`
Stream its logs

`argo retry <name>`
Rerun failed steps

`argo stop <name>`
Stop, run exit handlers

`argo delete <name>`
Remove a workflow

Manifest YAML file → `argo lint` → **Submit** creates a Workflow → `argo submit` → **Running** one pod per step → `argo logs` → **Finished** Succeeded or Failed → `argo retry` → **Rerun** same workflow

CONCEPTS

Use @latest instead of a workflow name to target the most recent run.

Reusable pieces live in WorkflowTemplates; submit one with --from workflowtemplate/<name>.

SUBMIT & LINT

`argo submit <file>`
Submit a workflow

`argo submit <file> --watch`
Submit and follow progress

`argo submit <file> -w`
Submit and wait for completion

`argo submit <file> --log`
Submit and tail logs

`argo submit <file> -p <name>=<value>`
Set an input parameter
Repeat -p for several parameters; -f <file> loads them from a file.

`argo submit <file> -f <params-file>`
Load parameters from a file

`argo submit --from workflowtemplate/<name>`
Submit from a template

`argo submit --from cronworkflow/<name>`
Run a cron workflow now

SUBMIT & LINT (cont.)

`argo submit <file> --entrypoint <template>`
Override the entrypoint

`argo submit <file> --serviceaccount <sa>`
Run pods as a service account

`argo submit <file> --dry-run -o yaml`
Preview the final manifest

`argo lint <file or dir>`
Validate manifests

`argo lint --offline <dir>`
Lint without a cluster

INSPECT

`argo list`
List workflows

`argo list -A`
List across all namespaces

`argo list --running`
Only running workflows

`argo list --completed`
Only completed workflows

`argo list --status Failed,Error`
Filter by status

`argo list -l <key>=<value>`
Filter by label

`argo list --since 1d`
Created in the last day

`argo list -o name`
Names only, for scripting

INSPECT (cont.)

`argo get <name>`
Show a workflow and its steps

`argo get @latest`
Show the latest workflow

`argo get <name> -o yaml`
Full manifest and status

`argo get <name> --status Failed`
Only nodes with a status

`argo watch <name>`
Watch until it completes

`argo wait <name>`
Block until it completes

LOGS

`argo logs <name>`
Print workflow logs

`argo logs <name> -f`
Stream logs

`argo logs @latest -f`
Follow the newest run

`argo logs <name> <pod>`
Logs of one pod

`argo logs <name> --tail <n>`
Last n lines

`argo logs <name> --since 10m`
Only recent lines

`argo logs <name> --grep <text>`
Show matching lines only

CONTROL

`argo retry <name>`
Retry failed steps

`argo retry <name> --restart-successful --node-field-selector <field>=<value>`
Also rerun chosen good nodes

`argo retry <name> --watch`
Retry and follow

`argo resubmit <name>`
Start a fresh copy

`argo resubmit <name> --memoized`
Reuse successful step outputs

`argo stop <name>`
Stop, letting exit handlers run

`argo terminate <name>`
Kill now, skip exit handlers

`argo suspend <name>`
Pause a workflow

`argo resume <name>`
Resume a paused workflow

DELETE

`argo delete <name>`
Delete a workflow

`argo delete --completed`
Delete all finished workflows

`argo delete --older 7d`
Delete those finished over 7 days ago

DELETE (cont.)

`argo delete --status Failed`
Delete by status

`argo delete --dry-run --completed`
Preview what would go

TEMPLATES

`argo template create <file>`
Create a WorkflowTemplate

`argo template list`
List WorkflowTemplates

`argo template get <name>`
Show a WorkflowTemplate

`argo template update <file>`
Update a WorkflowTemplate

`argo template delete <name>`
Delete a WorkflowTemplate

CRON

`argo cron create <file>`
Create a cron workflow

`argo cron list`
List cron workflows

`argo cron get <name>`
Show a cron workflow

`argo cron suspend <name>`
Pause the schedule

`argo cron resume <name>`
Resume the schedule

CONNECTION & SCRIPTING

`argo -n <namespace> <command>`
Pick the namespace

`argo --kubeconfig <path> <command>`
Use another kubeconfig

`argo --context <ctx> <command>`
Use another kube context

`argo -s <host:port> <command>`
Talk to an Argo Server

`argo submit <file> -o name`
Print only the new name

`argo version`
Print CLI version

RECIPES

Lint, submit, follow

```
argo lint wf.yaml
argo submit wf.yaml -n ci --watch
argo logs @latest -n ci -f
```

Fix a failed run

```
argo list --status Failed
argo get <name>
argo logs <name> --tail 50
argo retry <name> --watch
```

Clean up old runs

```
argo delete --older 7d --completed --dry-run
argo delete --older 7d --completed
```