

ArgoCon EU 2025

Argo Workflow Templates:

A Practical Deep-Dive

Tim Collins – Pipekit

Becky Pauley – Jetstack Consult



April 1, 2025

Intros



Tim Collins

Staff Infrastructure Engineer @ Pipekit.io

- Argo Maintainer. Lives in the CNCF Argo Slack
 - Member of the Pipekit Services team
- Can grow hair if he wants to. Just chooses not to, ok?



Becky Pauley

Senior Solutions Engineer @ Jetstack Consult

- Helping people solve K8s/Platform problems
 - Argo enthusiast
- My hobby is falling off things



Jetstack Consult



We help businesses, large and small, solve five key platform engineering challenges that will benefit their organization.



Platform
optimisation



Cloud
diversification



Infrastructure
modernisation



Innovation even
at scale



Security
by default

About Pipekit



Scale Argo & Kubernetes with Pipekit



Direct support from 40% of the active **Argo Workflows** maintainers in the world.



Save engineering time and up to 60% on compute costs



Add 3 **Argo** maintainers and 7 **Argo** contributors to your team



Serving **startups & Fortune 500** enterprises since 2021:

Enterprise Support for Argo:

Ideal for Platform Eng teams scaling with Argo

Control Plane for Argo Workflows:

Ideal for data teams, granular RBAC, and multi-cluster architectures



Image credit: Phil Botha, Unsplash

<https://pipek.it/argocon-templates-talk>

New Phone, Who 'Dis?

```
apiVersion: argoproj.io/v1alpha1
kind: Workflow
metadata:
  generateName: hello-world-
spec:
  entrypoint: hello-world
  templates:
    - name: hello-world
      container:
        image: busybox
        command: [echo]
        args: ["Hello ArgoCon!"]
```



Talk: Demystifying Argo Workflows
<https://pipek.it/argocon-demystifying-wf-talk>

Template Types



Invocators

Used to invoke/call other templates and provide execution control.

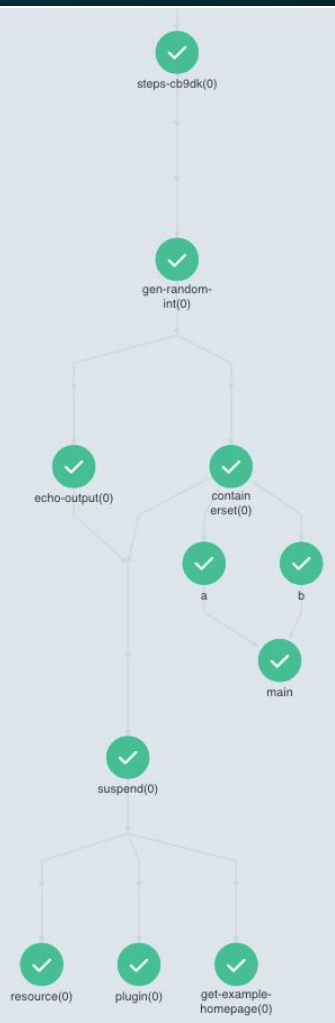


Definitions

Define work to be done, usually in a container.

Template Invocators: Steps

```
spec:
  entrypoint: main
  templates:
    - name: main
      steps:
        - name: gen-random-int
          template: gen-random-int
        - name: echo-output
          template: echo-output
        - name: containerset
          template: containerset
        - name: suspend
          template: suspend
        - name: get-example-homepage
          template: http
        - name: plugin
          template: plugin
        - name: resource
          template: resource
```



Template Invocators: Steps

Order of execution

- - - outer lists = sequential
- - inner lists = parallel
- Conditional:

```
# evaluate the result in parallel
- - name: heads
  template: heads
  when: "{{steps.flip-coin.outputs.result}}" == heads"
- name: tails
  template: tails
  when: "{{steps.flip-coin.outputs.result}}" == tails"
```

Template Invocators: DAG (Directed Acyclic Graph)

```
spec:
  entrypoint: main
  templates:
    - name: main
      dag:
        tasks:
          - name: gen-random-int
            template: gen-random-int
          - name: echo-output
            template: echo-output
            depends: gen-random-int
          - name: containerset
            template: containerset
            arguments:
              parameters:
                - name: hi
                  value: "Hello ArgoCon!"
            depends: gen-random-int
```

...



Template Invocators: Use cases



Steps:

- Simpler when starting out
- Deterministic Workflows
- Short Workflows



DAG:

- Flexibility and control
- Dependencies between Workflow tasks
- Prefer these - but ensure readability

Template Types



Invocators

Used to invoke/call other templates and provide execution control.



Definitions

Define work to be done, usually in a container.

Template Definition: **Container**



- The most common template type
- It will schedule a Pod to run our Container.
- The same as the Kubernetes container spec
- Automatic output of result as a parameter.

```
- name: hello-world
  container:
    image: busybox
    command: [echo]
    args: ["Hello ArgoCon!"]
```

Template Definition: **Script**



- A convenience wrapper around a **Container Template**.
- Allows you to define a script in place.
- Same automatic result output as Container Templates.

```
- name: gen-random-int
  script:
    image: python
    command: [python]
    source: |
      import random
      i = random.randint(1,100)
      print(i)
```



Image credit: Frank Mckenna, Unsplash

Template Definition: So far...



Definitions

Define work to be done,
usually in a container.

Container

Script

...

Template Definition: Container Set

```
containerSet:
  volumeMounts:
    - mountPath: /workspace
      name: workspace
  containers:
    - name: a
      image: alpine
      command:
        - sh
        - -c
        - |
          echo 'a: Hi ArgoCon!'
    - name: b
      image: alpine
      command:
        - sh
        - -c
        - |
          echo 'b: Hi ArgoCon!'
```

<https://argo-workflows.readthedocs.io/en/latest/container-set-template>



- Short-lived containers - reduce time and resource overhead.
- Can only include containers (not other template types e.g. script, resource).
- Multiple containers in the same pod – resource implications!



Image credit: Frank Mckenna, Unsplash

Template Definition: **Resource**



- Performs operations on cluster resources directly.
- Used to get, create, apply, delete, replace or patch resources in your cluster.

```
- name: create-a-configmap-please
  resource:
    action: create
    manifest: |
      apiVersion: v1
      kind: ConfigMap
      metadata:
        generateName: cm-
      data:
        some: value
```

Template Definition: Resource - Tips



```
spec:
  entrypoint: create-a-job-please
  templates:
    - name: create-a-job-please
      resource:
        action: create
        setOwnerReference: true
        successCondition: status.succeeded > 0
        failureCondition: status.failed > 3
      manifest: |
        apiVersion: batch/v1
        kind: Job
        labels:
          workflows.argoproj.io/workflow: "{{workflow.name}}"
```

Continued...

Template Definition: Resource - Tips

Workflow of workflows

```
- name: create-a-workflow-please
  resource:
    action: create
    successCondition: status.phase == Succeeded
    failureCondition: status.phase in (Failed, Error)
    manifest: |
      apiVersion: argoproj.io/v1alpha1
      kind: Workflow
      metadata:
        generateName: workflow-of-workflows-1-
      spec:
        workflowTemplateRef:
          name: argocon
```

Template Definition: Resource - Tips



RBAC !





Image credit: Frank Mckenna, Unsplash

Template Definition: **Suspend**



- Suspends workflow execution.
 - Either for a duration
 - Or until resumed manually
- Resume via the CLI (argo resume), the API endpoint, or the UI

```
- name: delay
  suspend:
    duration: "20s"
```


Template Definition: So far...



Definitions

Define work to be done,
usually in a container.

Container

Script

ContainerSet

Resource

Suspend

Template Definition: HTTP/Plugin

Agent pod



One per Workflow

Used/reused by HTTP
and Plugin steps/tasks

Need extra
permissions!

<https://argo-workflows.readthedocs.io/en/latest/http-template/#argo-agent>

Template Definition: HTTP

- name: http

http:

url: https://example.com/

method: "GET"

<https://www.youtube.com/watch?v=tohzCqpog5A>



Template Definition: **Plugin**

```
- name: main
  plugin:
    hello: { }
```



- Runs an executor plugin
- Source for plugin:



<https://github.com/argoproj-labs/argo-workflows-hello-executor-plugin>

HTTP and Plugin Template Definitions

USE WHEN

- They meet your use-case
- You will be calling them multiple times in the same workflow
- They are replacing short-lived steps that would otherwise be their own pod

Efficiency/reuse

DISADVANTAGES

- Extra setup and RBAC
- Harder to reason about (more abstraction)

Use with caution - can another template type fit my use-case?

Making your life easier: **Template choice**



Definitions

Define work to be done,
usually in a container.

Container

Resource

Script

HTTP

ContainerSet

Suspend

Plugin

Template Invocator Invocator: **Inline**

```
- name: main
  dag:
    tasks:
      - name: a
        inline:
          container:
            image: alpine
            command:
              - sh
              - -c
              - |
                echo "Hi!"
```



Making your life easier: **templateDefaults**

```
spec:
  entrypoint: main
  templateDefaults:
    timeout: 30s    # timeout value will be applied to all templates
    retryStrategy: # retryStrategy value will be applied to all templates
      limit: "2"
```

- Can set in:
 - Workflow-controller-configmap
 - Workflow
- Override in individual templates when defaults don't suit

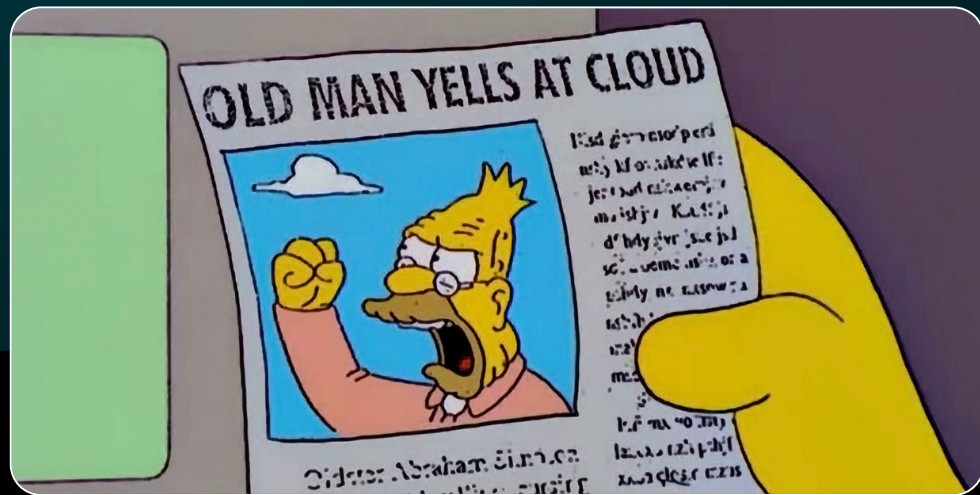



```
grumble@argocon:~ $ argo template list
```

```
error="workflow template foo not found"
```

Other things that have the word 'template' in them but aren't anything to do with the templates we have been talking about.

- WorkflowTemplate
- TemplateRef
- ClusterWorkflowTemplate



```
templates:
- name: aaargh
  steps:
    - name: sigh
      templateRef: # This is how you reference a workflowTemplate (one word)
        name: hello-world # This is the workflow template (with a space)
```

Want Change?



<https://github.com/argoproj/argo-workflows/pull/13987>

<https://github.com/argoproj/argo-workflows/pull/14110>

<https://github.com/argoproj/argo-workflows/pull/14143>

<https://github.com/argoproj/argo-workflows/pull/14198>

<https://github.com/argoproj/argo-workflows/pull/14253>

<https://github.com/argoproj/argo-workflows/pull/14292>



[**https://pipek.it/argo-templates-survey**](https://pipek.it/argo-templates-survey)

Stuff!

This talk & examples:



<https://pipek.it/argocon-templates-talk>

Free Argo/Infrastructure Help & Advice:



Booth S590 - Pipekit

Booth N763 - CyberArk (Jetstack Consult)



<https://pipek.it/argo-templates-survey>