Native k8s CI/CD Pipeline

Shash Reddy (@shashwathithi)
What is CI?

Commit changes → Source repository → CI server
Build → Test
Success/Fail
Results
What is CD?

Commit changes

Source repository

CI server

Build

Test

Success/Fail

Commit changes

Results

Results

Deploy

Production
WHY?
Why K8s?

- Containerization
- Orchestration
- Observability
- Self healing system
- Cloud platform agnostic
- Identity management
HOW?
- Custom Resources
- Custom Controllers
WHAAATT?
PipelineResource CRD

PipelineResources

types

Image

Github

Cloud storage

...
apiVersion: tekton.dev/v1alpha1
kind: PipelineResource
metadata:
  name: myapp-source

spec:
  type: git

  params:
  - name: revision
    value: master
  - name: url
    value: https://github.com/shashwathi/helloworld-go.git
apiVersion: tekton.dev/v1alpha1
kind: PipelineResource
metadata:
  name: myapp-source
spec:
  type: git

params:
  - name: revision
    value: master
  - name: url
    value: https://github.com/shashwathi/helloworld-go.git
Task CRD

Task

Composes

Input Resource

Step 1

Executes steps in serial

Step n

Output Resource
EXAMPLE

Source-repo \rightarrow Build-app-image \rightarrow Docker-image
spec:
  inputs:
    params:
      - name: DOCKERFILE
description: Path to the Dockerfile.
default: ./]Dockerfile
  resources:
    - name: source
type: git

outputs:
  resources:
    - name: appImage
type: image
steps:
  - name: build-and-push
    workingdir: ${inputs.resources.source.path}
    image: gcr.io/kaniko-project/executor:v0.9.0
    command:
      - /kaniko/executor
      - --dockerfile=${inputs.params.DOCKERFILE}
      - --destination-path=${outputs.resources.appImage.url}
```yaml
spec:
  inputs:
    params:
      - name: DOCKERFILE
        description: Path to the Dockerfile.
        default: ./Dockerfile
    resources:
      - name: source
        type: git

outputs:
  resources:
    - name: appImage
      type: image

steps:
  - name: build-and-push
    workingdir: ${inputs.resources.source.path}
    image: gcr.io/kaniko-projectexecutor:v0.9.0
    command:
      - /kaniko/executor
      - --dockerfile=${inputs.params.Dockerfile}
      - --destination-path=${outputs.resources.appImage.url}
```

spec:
  inputs:
    params:
      - name: DOCKERFILE
        description: Path to the Dockerfile.
        default: ./Dockerfile
  resources:
    - name: source
      type: git
  outputs:
    resources:
      - name: appImage
        type: image

steps:
- name: build-and-push
  workingdir: ${inputs.resources.source.path}
  image: gcr.io/kaniko-project/executor:v0.9.0
  command:
    - /kaniko/executor
    - --dockerfile=${inputs.params.DOCKERFILE}
    - --destination-path=${outputs.resources.appImage.url}
Taskrun CRD

- Task
  - Input PipelineResource
  - Task
  - Output PipelineResource

- Pod
  - Service Account
  - Input PipelineResource
  - Step 1
  - Step n
  - Shared Volume/workspace
  - Output PipelineResource

- Creates

- Shared Volume
  - /workspace
spec:
  taskRef:
    name: kaniko
    inputs:
      resources:
        - name: source
          resourceRef:
            name: myapp-source
    outputs:
      resources:
        - name: app-image
          resourceRef:
            name: myapp-image
spec:
  taskRef:
    name: kaniko

inputs:
  resources:
    - name: source
      resourceRef:
        name: myapp-source

outputs:
  resources:
    - name: app-image
      resourceRef:
        name: myapp-image

spec:
  inputs:
    params: ...
    resources:
      - name: source
        type: git

outputs:
  resources:
    - name: app-image
      type: image

steps:
  name: build-and-push
  ...
spec:
  taskRef:
    name: kaniko
  inputs:
    resources:
      - name: source
        resourceRef:
          name: myapp-source
  outputs:
    resources:
      - name: app-image
        resourceRef:
          name: myapp-image

spec:
  steps:
    ...
  inputs:
    params:
      resources:
        - name: source
          type: git
  outputs:
    resources:
      - name: app-image
        type: image
Summary

Task CRD

Validates against

TaskRun

1. Get resource

Input PipelineResource

2. Create

Pod to execute steps

3. Update resource

Output PipelineResource

Kubectl apply -f taskrun.yaml

Credit to @nader-ziada
Pipeline CRD

Pipeline

Input PipelineResource

Task 1

::

Task n

Output PipelineResource

Composes

Executes tasks in serial/parallel
spec:

  resources:
  - name: source-repo
    type: git
  - name: app-image
    type: image

  tasks:
  ...
spec:
  resources:
  - name: source-repo
    type: git
  - name: app-image
    type: image

tasks:
- name: run-unit-tests
  taskRef:
    name: unit-tests
  resources:
    inputs:
    - name: workspace
      resource: source-repo
    - name: build-app-image
  runAfter: [run-unit-tests]
  taskRef:
    name: kaniko

Task for unit tests
- name: run-unit-tests
  taskRef:
    name: unit-tests
  resources:
    inputs:
      - name: workspace
        resource: source-repo

- name: build-app-image
  runAfter: [run-unit-tests]
  taskRef:
    name: kaniko
  params:
    ...
  resources:
    inputs:
      ...
    outputs:
      ...
Pipelinerun CRD

- **Input PipelineResource**
- **Pipeline**
- **Output PipelineResource**

**Creates**

- Taskrun 1
- Taskrun 2
- \(\cdots\)
- Taskrun \(N\)
apiVersion: tekton.dev/v1alpha1
kind: PipelineRun
metadata:
  name: demo-pipeline-run

spec:
  pipelineRef:
    name: demo-pipeline

    serviceAccount: default
    resources:
      - name: source-repo
        resourceRef:
          name: myapp-git-source
      - name: app-image
        resourceRef:
          name: myapp-docker-image
apiVersion: tekton.dev/v1alpha1
kind: PipelineRun
metadata:
  name: demo-pipeline-run
spec:
  pipelineRef:
    name: demo-pipeline
  serviceAccount: default
  resources:
  - name: source-repo
    resourceRef:
      name: myapp-git-source
  - name: app-image
    resourceRef:
      name: myapp-docker-image

Credentials
apiVersion: tekton.dev/v1alpha1
kind: PipelineRun
metadata:
  name: demo-pipeline-run
spec:
pipelineRef:
  name: demo-pipeline
serviceAccount: default
resources:
  - name: source-repo
    resourceRef:
      name: myapp-git-source
  - name: app-image
    resourceRef:
      name: myapp-docker-image

Binding Pipeline resources to runtime object
Kubectl apply -f pipelinerun.yaml

Pipeline CRD

Validates against

PipelineRun

Creates 1.* taskrun(s)

TaskRun

PVC/Cloud Storage Bucket

• GET Input resources from previous tasks
• PUT Output resources

Create PVC/bucket to share artifacts between tasks

Credit to @nader-ziada
References

- [Tektoncd Pipeline](#) project
- [Documentation](#)
- [Catalog of tasks](#)
- [Roadmap](#) document
- [Contributors guide](#) to get started