Using epic-app with Substrate

Document Level Classification

200

- Further Examples

Introduction

epic-app is an opinionated Helm chart offered to all teams across Epic to abstract away typical resources and configurations for workloads running on Substrate infrastructure. When epic-app is added as a dependency to a new or pre-existing Helm chart, it allows you to extend the functionality of your app using the already existing features of epic-app.

Why use epic-app and what value does it provide?

In essence, epic-app is a pre-written Helm chart with out of the box features available to you. Using the epic-app Helm chart allows you to create a new application quickly without having to write the code from the ground up every time. Epic app is also updated frequently to add new features and/or functionality. Upgrading to a new version gives you the benefit of added functionality without having to write the code for that functionality yourself. This is covered in the Upgrading Versions section below.

What's included?

The epic-app Helm chart comes with a values.yaml file. This file can be copied for use in your chart or you can use your own values.yaml file referencing epic-app (more about this file below). When epic-app is referenced in your chart, you can customize input values for your app. Based on your customizations, templates are used for the desired functionality. These templates are already written so you don't have to spend time creating your own. Those templates include the following:

Template	Purpose

canary-deployment.yaml	Canary Deployments confi
canary-service.yaml	Used with Canary Deployn
configmap.yaml	Configuration data
external-secret.yaml	Vault secrets injection with
daemonset.yaml	Configuration of pods per
deployment.yaml	Deployment Configuration
hpa.yaml	Horizontal Pod Autoscaling
ingress.yaml	Ingress configuration
keda-scaledobjects.yaml	Kubernetes Even Driven A
keda-trigger-authentications.yaml	Authentication for KEDA
overriden_ingress.yaml	Override of default ingress

pdb.yaml	Pod distribution budget co
service.yaml	Service configuration
serviceaccount.yaml	ServiceAccount configurati

What are the default values for epic-app?

Included with epic-app is the values.yaml file. This Helm chart values manifest, as-is, contains the default values. To utilize epic-app in your application, you would add epic-app as a dependency (See "**Add epic-app as a dependency**" below). After this is added as a dependency, the repo should be added as an available Helm repo (See "**Add the Helm repo**" below). Now that epic-app has been added as a dependency and the Helm repo is added to the list of available repos, the epic-app key is referenced in your chart along with configuration values you want to pass to epic-app (See "**Adding Configuration values to values.yaml**" below). Essentially when you pass configuration values to epic-app, you are overriding the default values so your application uses only the features you specify as values.

For a complete list of all of the values and their defaults, refer to the README.md in the epic-app repo.

Prerequisites

- **Toolchain Setup**
  - Complete the Prerequisites to ensure you have the appropriate access and local developer toolchain setup.

- **Artifactory Access Token**
    - Head to Artifactory and log in.
    - In the top right corner, click your username and then click ***Edit Profile.***
    - Generate an identity token and save it somewhere safe.

Adding epic-app to your Helm chart

01 - Create a new Helm chart

From your command prompt run the following commands to create a new Helm chart:

```
$ helm create example-chart
$ tree
\---example-chart
    |   .helmignore
    |   Chart.yaml
    |   values.yaml
    |
    +---charts
    \---templates
        |   deployment.yaml
        |   hpa.yaml
        |   ingress.yaml
        |   NOTES.txt
        |   service.yaml
        |   serviceaccount.yaml
        |   _helpers.tpl
        |
        \---tests
```

```
                test-connection.yaml
$ cd example-chart
```

02 - Add the *epic-app* as a dependency

In the `Chart.yaml` file add the following section:

```
dependencies:
- name: epic-app
  version: some_version  # ~some_version would also be acceptable e.g. ~
  repository: @substr-helm
```

Example `Chart.yaml`

```
##############################################################################
# HELM CHART FOR `example-chart`
##############################################################################

#
# Reference:
# * https://helm.sh/docs/topics/charts/#the-chartyaml-file
#
apiVersion: v2
name: example-chart
version: "1.0.0-<USERNAME_ALNUM>" # Example 1.0.0-user100
description: A Helm chart for Kubernetes

# Add a dependency on epic-app (https://github.ol.epicgames.net/charts/
dependencies:
  - name: epic-app
    repository: @substr-helm
    # epic-app versions and associated release notes is available here:
```

```
    # https://github.ol.epicgames.net/charts/epic-app/releases
    version: "~3.1.0"
```

Additional information on Helm `Chart.yaml` file, refer to the Helm Charts documentation.

03 - Add the Helm repo

From a command prompt run the following commands to list the available repos and add the substrate Helm repo if it is not already available:

```
# List available Helm repositories
helm repo list

# If not listed above, add the "substr-helm" repository hosted on Artif
helm repo add substr-helm https://artifacts.ol.epicgames.net/artifactor
```

Replace the `<USERNAME>` tag with your Artifactory user name and `<ARTIFACTORY_ACCESS_TOKEN>` with the access token created in the Prerequisites section.

04 - Add Configuration values to values.yaml

Add the `epic-app:` key to the `values.yaml` file as seen in the example below.

```
epic-app:
  #
  # Reference:
  # * https://github.ol.epicgames.net/charts/epic-app/blob/main/values.
  #
```

Example `values.yaml`

```
##################################################################
# Configuration values for example-chart
##################################################################

#
# NOTE:
# These are values passed to the epic-app Chart, and therefore must be
# under a top level property that matches the `dependencies[].name` in
# `Chart.yaml`. By convention, the dependency name is also `epic-app`.
#
epic-app:
  #
  # Reference:
  # * https://github.ol.epicgames.net/charts/epic-app/blob/main/values.
  #
  resourceTags:
        # Required tags. https://confluence-epicgames.atlassian.net/wik
    service: "<name of the service>"
    owner: "<name of the owner>"
    contact: "<email of the owner>"
    euid: "<EUID in Epic's service catalog>"
  arm:
    enabled: true # This requires ARM64 container builds to be created
  containers: # This is a map of containers that are in the Pod
    # Application container
    example-chart:
      image:
        name: "artifacts.ol.epicgames.net/<DOCKER_REPO>/example-chart"
        tag: "1.0.0-<USERNAME_ALNUM>" Example 1.0.0-user100
      ports:
        - 80
      environment:
```

```
        LOG_LEVEL: INFO

##########################################################################
```

05 - Update Helm chart dependencies

From a command prompt run the following command to update the Helm chart's dependencies:

```
$ helm dependency update
```

Upgrading epic-app Versions

For a detailed walkthrough of upgrades and upgrade scenarios, refer to Upgrading epic-app to a new version

Using Features Not Supported by epic-app

Sometimes developers want to use a Kubernetes feature that isn't officially supported in Epic App. In these cases it will generally depend on how quickly this feature needs to be implemented. Below are three different ways to use a feature that is unsupported by Epic App out of the box.

Creating a PR for a Feature Not supported by epic-app

In this scenario a developer would be interested in contributing to the development of epic-app by creating a Pull Request for a new feature. The process of submitting a PR can take some time as it involves discussion with epic-app developers and requires approval. It's recommended to reach out in #cloud-ops-support-ext before opening up a PR to initiate discussion and receive advice regarding features that could be included.

Using Custom Templates for a Feature Not Supported by epic-app

If the feature you need to use is either not accepted by the epic-app devs or you need to start using a feature sooner than the time it takes to submit a PR and go through the entire process, you can use custom templates for resources that don't exist in epic-app by default. For example, if you wanted to include the `Job` resource, you would create your own chart to serve as the Parent chart that uses epic-app as a dependency. Then you would create `templates/` folder under the `deploy/` folder where you would include any custom YAML so that when it's rendered, it'll include the extra resources.

Example Directory Structure

```
parent-chart                          <-------- Parent Chart
|   Dockerfile
├──deploy
|   └──chart
|       |   Chart.yaml
|       |   values.yaml
|       |
|       ├──charts                     <-------- Dependency Charts
|       |       epic-app-3.4.2.tgz
```

```
|       |
|       └──templates                  <-------- Custom Templates
```

For documentation on templates and best practices, refer to the Helm Chart Developers Guide and Helm Template Best Practices.

Using Umbrella Charts

More information on this in the dedicated umbrella document.

Further Examples

For more examples of how to use epic-app in your applications, refer to Deploying your first application to Substrate. You can also find examples of specific sections of the values.yaml file in the repo by looking at the values.md. Note, some sections may not include an example.

Contributing

If you have a new feature that you think would benefit Epic App, refer to the Contributing section of the epic-app repo here https://github.ol.epicgames.net/charts/epic-app.

- Upgrading epic-app to a new version
- Using Umbrella Charts with epic-app

Page Information:
Page ID: 81068258