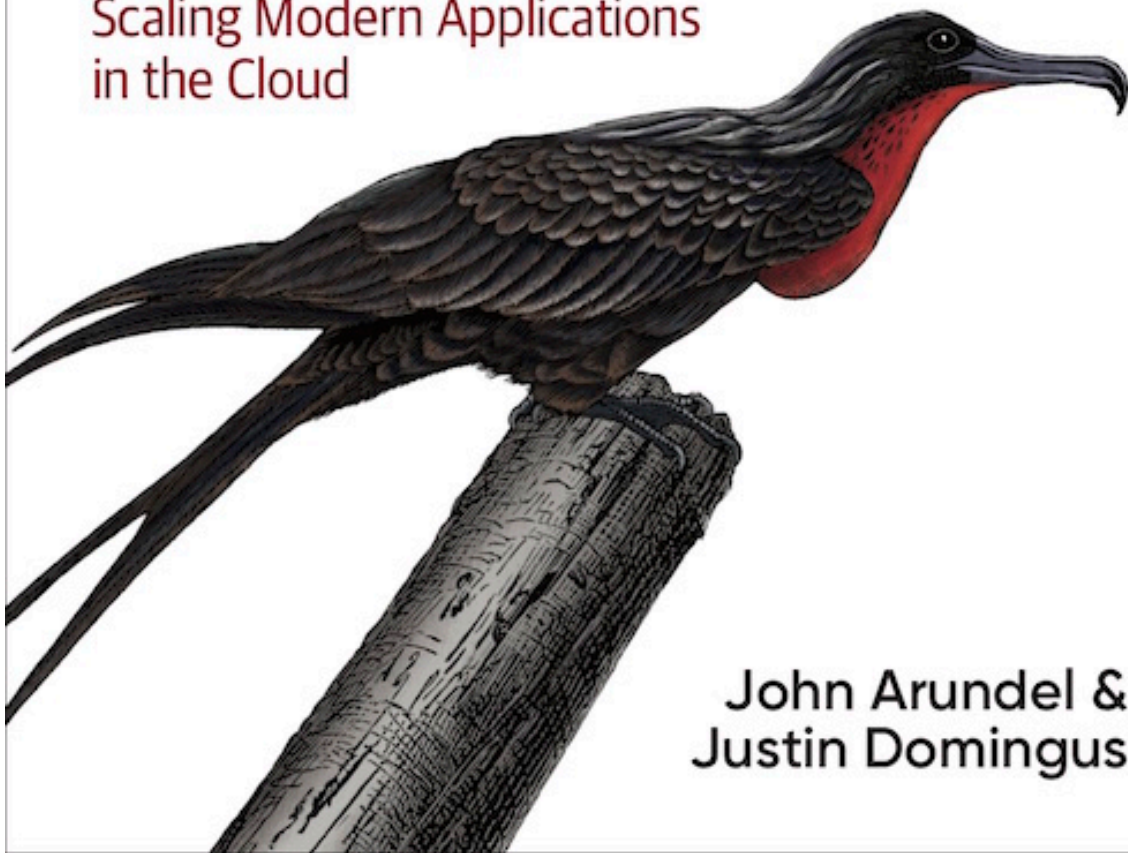

O'REILLY®

Cloud Native DevOps with Kubernetes

Building, Deploying, and
Scaling Modern Applications
in the Cloud



John Arundel &
Justin Domingus

Welcome! This is the example code repository to accompany the book ‘Cloud Native DevOps with Kubernetes’, by John Arundel and Justin Domingus. Buy the book here:

- Amazon US
- Amazon UK
- O’Reilly

About the book

From the preface:

You’ll learn what Kubernetes is, where it comes from, and what it means for the future of software development and operations. You’ll learn how containers work, how to build and manage them, and how to design cloud native services and infrastructure.

You’ll understand the trade-offs between building and hosting Kubernetes clusters yourself, and using managed services. You’ll learn the capabilities, limitations, and pros and cons of popular Kubernetes installation tools such as kops, [kubeadm](#), and Kubespray. You’ll get an informed overview of the major managed Kubernetes offerings from the likes of Amazon, Google, and Microsoft.

You’ll get hands-on practical experience of writing and deploying Kubernetes applications, configuring and operating Kubernetes clusters, and automating cloud infrastructure and deployments with tools such as Helm. You’ll learn about Kubernetes support for security, authentication, and permissions, including Role-Based Access Control (RBAC), and best practices for securing containers and Kubernetes in production.

You’ll learn how to set up continuous integration and deployment with Kubernetes, how to back up and restore data, how to test your cluster for conformance and reliability, how to monitor, trace, log, and aggregate metrics, and how to make your Kubernetes infrastructure scalable, resilient, and cost-effective.

The book aims to teach you everything you need to know to deploy, run, and scale applications in Kubernetes, and most importantly, to give you working example code for everything we demonstrate. That code is open source, available for free for you to use and adapt whether or not you buy the book. And here it is!

Show me the code

Almost all the example code involves our ‘hello world’ demo application. Here is the list of examples; follow the links to see the documentation on each example.

- Build and run the demo application locally (start here!)
- Deploy the app to Kubernetes using kubectl
- Deploy the app with Helm (or with Helm 3)
- Manage the app with Helmfile
- Set up a namespace, resource requests, and limits for the app
- Add config data to the app’s environment
- Pass config data to the app’s command line
- Mount a config file in the container at runtime
- Add secret data to the app’s environment
- Mount a secrets file in the container
- Store encrypted secrets in the app repo, decrypted automatically on deploy
- Develop the app locally with Skaffold
- Build and deploy the app automatically with Drone
- Build and deploy the app automatically with Google Cloud Build

Terraform examples

We also include some Terraform code examples, to help you manage cloud resources with code. Unfortunately we didn’t have space to discuss these in the book, but we hope they’ll be useful to you anyway.

Google Cloud

- Create a cloud storage bucket in code
- Create a cloud database instance in code
- Create a Kubernetes cluster in code

Amazon AWS

- Create a cloud storage bucket in code
- Create a cloud database instance in code
- Create a cloud virtual machine instance in code

You will need

To build and run all of these examples, you will need:

- Go (any recent version is fine)
- Docker version 18.03 or above

Where you need other tools for specific examples, we'll mention that in the README for the example.

Contributing to the repo

We would absolutely love it if you contributed! Feel free to send us a PR to add new examples, add versions of the examples for different cloud providers (for example Microsoft Azure), or fix or improve the existing examples.

Known Issues

apiVersion

When we released the book most k8s clusters still used `extensions/v1beta1` for Deployments. On newer versions of k8s `Deployment` has been moved to `apiVersion: apps/v1`. If you get the message:

```
1 error: unable to recognize "k8s/deployment.yaml": no matches for kind "Deployment" in version "extensions/v1beta1"
```

when trying out the examples then try updating `apiVersion: extensions/v1beta1` to `apiVersion: apps/v1` in your `deployment.yaml` file(s).

Thanks to @thescott for pointing this out.

Helm 3

At the time of publishing Helm 3 was not yet released and we included the Helm 2 examples that included the additional steps of installing `tiller`.

Helm 3 is now released and `tiller` is no longer required.

We have added the `hello-helm3` examples here with updated instructions if you would like to use the latest version of Helm.

You can read more about the changes between version 2 and 3 here.

kubectl run and Pod/Deployment v1.18 change

In the 1.18 release of Kubernetes the `kubectl run` command changed from creating a `Deployment` by default to creating a `Pod` instead.

We have a few examples where we use `kubectl run` to get familiar with running a container in k8s.

Later we discuss why using the declarative `kubectl apply -f...` is preferred over the imperative `create`, `edit` or `run`, because your version-controlled YAML files always reflect the real state of the cluster.

In our `kubectl run` example we show the output as `deployment.apps "demo" created` but on version 1.18 instead you will instead see `pod/demo created`.

The subsequent port-forward example would instead be: `kubectl port-forward pod/demo 9999:8888`

Again, using `kubectl apply -f...` and keeping your manifests tracked in source control is a better long-term solution.

Service port 8888 VS 9999

Depending on which version of the book you read you may see reference to using port 9999 for the service port here. This caused some confusion between the pod port, service port, and the port-forwarding port in the examples, so was changed in the latest revision to use 8888 for both the pod and the service. Thanks to @randoljt for catching this and sorry for any confusion.