O'REILLY®

Covers OpenShift 3

OpenShift for Developers

A GUIDE FOR IMPATIENT BEGINNERS

Grant Shipley & Graham Dumpleton

O'REILLY®

OpenShift for Developers

Keen to build web applications for the cloud? Get a quick hands-on introduction to OpenShift, the open source Platform as a Service (PaaS) offering from Red Hat. With this practical guide, you'll learn the steps necessary to build, deploy, and host a complete real-world application on OpenShift without having to slog through long, detailed explanations of the technologies involved.

OpenShift enables you to use Docker application containers and the Kubernetes cluster manager to automate the way you create, ship, and run applications. Through the course of the book, you'll learn how to use OpenShift and the WildFly application server to build and then immediately deploy a Java application online.

- Learn about OpenShift's core technology, including Docker-based containers and Kubernetes
- Use a virtual machine with OpenShift installed and configured on your local environment
- Create and deploy your first application on the OpenShift platform
- Add language runtime dependencies and connect to a database
- Trigger an automatic rebuild and redeployment when you push changes to the repository
- Create and build Docker-based images for your application

Grant Shipley is a Senior Manager at Red Hat and a Developer Advocate for OpenShift. He has over 15 years of software development experience with Java and PHP. Grant also contributes to open source projects and builds mobile applications.

Graham Dumpleton is a Developer Advocate for OpenShift at Red Hat. He's an active member of the Python software developer community and the author of mod_wsgi, a popular module for hosting Python web applications in conjunction with the Apache HTTPD web server.

"*OpenShift for Developers* is a well structured and detailed guide for facilitating OpenShift right from the ground up. The book is destined to be an authoritative reference for anyone getting acquainted with OpenShift and is specifically useful for developers, operations, and for infrastructure and solution architects."

—Ajith Joseph
Cloud Architect,
Intellect Design Arena, Inc.

WEB DEVELOPMENT

US $29.99 CAN $34.99
ISBN: 978-1-491-96143-8

Twitter: @oreillymedia
facebook.com/oreilly

OpenShift for Developers
by Grant Shipley and Graham Dumpleton

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://safaribooksonline.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Editors: Brian MacDonald and Heather Scherer
Production Editor: Melanie Yarbrough
Copyeditor: Christina Edwards
Interior Designer: David Futato
Cover Designer: Randy Comer
Illustrator: Rebecca Demarest

June 2016: First Edition

Revision History for the First Edition
2016-05-18: First Release

978-1-491-94300-7

[LSI]

Table of Contents

CHAPTER 1

Introduction to a Container Application Platform

In the few years since the original version of OpenShift was released, the cloud ecosystem has expanded at a rapid pace. New technologies and systems are springing up almost overnight. The new version of OpenShift (version 3) includes other technologies at the core of the platform, and it's worth it to spend some time learning about them and how to incorporate them into your development work. The core technology used as the basis of OpenShift includes Docker-based containers and orchestration via the Kubernetes system. Given that the core of the platform is based on containers, we often refer to OpenShift as a container application platform in that it is a platform designed for the development and deployment of containers.

So why did the OpenShift team rewrite their perfectly good PaaS (OpenShift 2)? The core architecture of OpenShift was built a bit over 4 years ago, which in the cloud evolution years (like dog years) is about 28 years ago, and a lot has changed since then. One example is Docker, which we discuss later in this chapter. The beginnings of the Docker project that you know today started as a container implementation using cgroups and kernel namespaces at a PaaS company named dotCloud. In March of 2013, the Docker project was released as an open source project and in July 2013, the company behind dotCloud pivoted and announced that their primary focus going forward would be the Docker container technology. OpenShift has been using containers since the beginning but the OpenShift team saw great potential within the Docker ecosystem—i.e., its potential as a "standard" for packaging applications, thereby creating better portability across environments.

The landscape in the cloud area has also been changing rapidly. The OpenShift team has four years of experience running one of the largest public PaaS systems and a large and successful install base of the on-premise enterprise version. The OpenShift team realized the time was right to use our knowledge, industry advancements, and new FOSS cloud projects to create an even better platform, which has been released as the OpenShift 3 container application platform.

Docker

Unless you have been living under a rock last two years, you have certainly heard about Docker and probably even heard about how Docker-based containers can solve all of your problems. While it is true that Docker-based containers are certainly cool, we need to be realistic about what this great technology provides and what it doesn't. In its simplest form, Docker provides users with a lightweight portable format that can be used to ship images of an application around to different environments. It accomplishes this while also packaging up all of the dependencies at both the system and application level to ensure the application runs as expected when deployed.

While Docker provides a great portable container format to ship applications around, it is important to remember that using Docker-based containers is just a small piece in the overall deployment puzzle that many organizations and individual developers face. A single Docker container that contains your application code is pretty trivial to get started with but once you move past the single container phase for your application, things become complex quickly, often leaving the developer to understand operational tasks instead of focusing on code. You may be thinking, "What on earth is he talking about?" For starters, things like:

- Load balancing a set of containers with or without session affinity
- Mounting persistent storage inside of the containers
- Placement and scheduling of containers on the infrastructure
- Rolling deployments and other operational considerations that traditional developers are not experts on

As you can see, things get complicated quickly when moving from a single development container to a real production application.

Given that Docker-based containers are such an important piece of the puzzle in a container application platform, Red Hat and the OpenShift team have spent a tremendous amount of engineering effort working on and with the Docker upstream project. In fact, at the time of writing, Red Hat is the second largest contributor to the project.

Kubernetes to the Rescue?

While Docker gets us a portable lightweight runtime, it lacks features for supporting n-tier applications. The OpenShift team understood this early on in the research phase when planning the new platform. We searched high and low for a great orchestration and scheduling system and made an early bet on the Kubernetes project Google started, and are now seeing great benefits as a result of that decision. Let's talk a little bit more about the Kubernetes project as you may not be familiar with it.

Kubernetes is a free and open source project started by Google in 2014. The first version was released in July 2015. The project is based off the Borg project, the technology Google uses to run containers at scale internally, launching over 7000 containers per second. When it came time to build their next version of the software they decided to open source the effort and Kubernetes was born.

Red Hat and the OpenShift team were key contributors to Kubernetes because of the value we see in the project to help with the scheduling, orchestration, and running of Docker-based containers for production workloads.

Again, just to be clear, both Docker and Kubernetes are great. However, they are just two pieces of the container application platform puzzle.

OpenShift

After all this goodness with Docker and Kubernetes, you are probably wondering where OpenShift fits in to the overall puzzle and what piece the platform provides. It turns out that Kubernetes is excellent at orchestrating and scheduling containers, but to have a platform that helps developers and sys admins deal with the piece most

important to their "customers"—the application—something more is needed. The goal of OpenShift is to provide the best experience for developers and sysadmins developing, deploying, and running applications. In other words, OpenShift is a layer on top of Docker and Kubernetes that makes it accessible and easy for the developer to create applications and a platform that is a dream for operators to deploy containers on for both development and production workloads.

We are going to quickly cover some of the features OpenShift brings to the table to help developers (since that is the focus of this book) and then we will explore these more in depth throughout the book.

Web-Based Console

The OpenShift platform ships with a feature-rich web console that allows developers to perform the actions needed to deploy and run existing source code projects. For example, one of the features (as shown in Figure 1-1), is a graphical representation of an application that consists of multiple containers all load balanced, while also using a database as the backend storage engine.

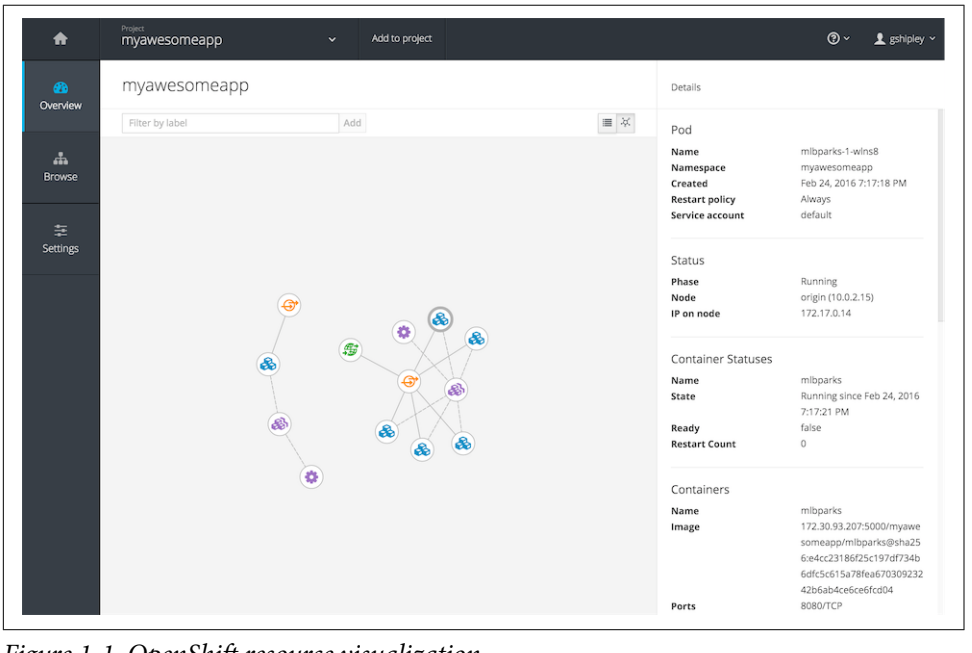

Figure 1-1. OpenShift resource visualization

Other tasks a developer would normally perform via the web console is scaling the application containers, creating projects, viewing log files, viewing the memory and CPU utilization of a container, and other common functions. The web console is a great tool as it provides a single interface for all aspects of your development project.

Command-Line Tool

OpenShift also provides a command-line tool that is written in the Go programming language. This tool, called *oc*, is a single binary executable provided for all major operating systems including Microsoft Windows, Apple OS X, and Linux. If you enjoy working on the command line, the *oc* tool is a first-class citizen and can be used to perform any operation that can be accomplished via the web console. The greatest benefit of using the `oc` command-line tool is that you have a single executable to perform all operations instead of having to interface directly with multiple tools such as the ones provides with Docker and Kubernetes. You can think of it as "one tool to rule them all." Okay, that may be a bad LOTR reference, I know.

Source-to-Image

The true power of OpenShift comes in with the S2I (Source-to-Image) open source project that the OpenShift Team created as part of the OpenShift platform. The team knows that developers want to take advantage of all of the benefits that running applications in containers provides, but don't want to spend their day creating Dockerfiles or running Docker builds while also having to do all of the orchestration by hand.

In its simplest form, the S2I-based Docker images allow developers to interact with the platform from a pure source code perspective. What this means is that OpenShift only needs to know the URL of your git repository and then the platform takes care of the rest. Under the covers, when you create a new project and container, the platform matches a base image of the desired runtime up with your source, performs a build, and then creates a new Docker image on the fly.

For a more concrete example, let's assume we have a Java-based project that uses the Maven build system. When using the WildFly application server and a git repository that contains your code, the platform will download the base WildFly image, clone your repo, identify it as a Maven project, run a Maven build, and take the artifact of that build and create a new Docker image that contains both the WildFly runtime and your build artifact. Finally, it will deploy the build artifact and then start up the resulting container. I know this may sound complicated, but don't worry, the platform takes care of all of the heavy lifting. We will dive into more detail as we progress through the book.

Integrated Logging and Metrics

One of the most important aspects of any software development project is the ability to troubleshoot issues as they arise. Since the beginning of time, one of the first places a developer looks when beginning the debug process is the application logs for the project. OpenShift provides a comprehensive view of your application logs including runtime logs, build logs, and deployment logs. Having these at your fingertips

through the web console and *oc* tool greatly simplifies the troubleshooting process when using containers.

Another important aspect is metrics about your applications, utilization of system resources. The OpenShift platform includes support for this as shown in Figure 1-2, which details the memory and CPU utilization for a container running a Java application server and accompanying application.

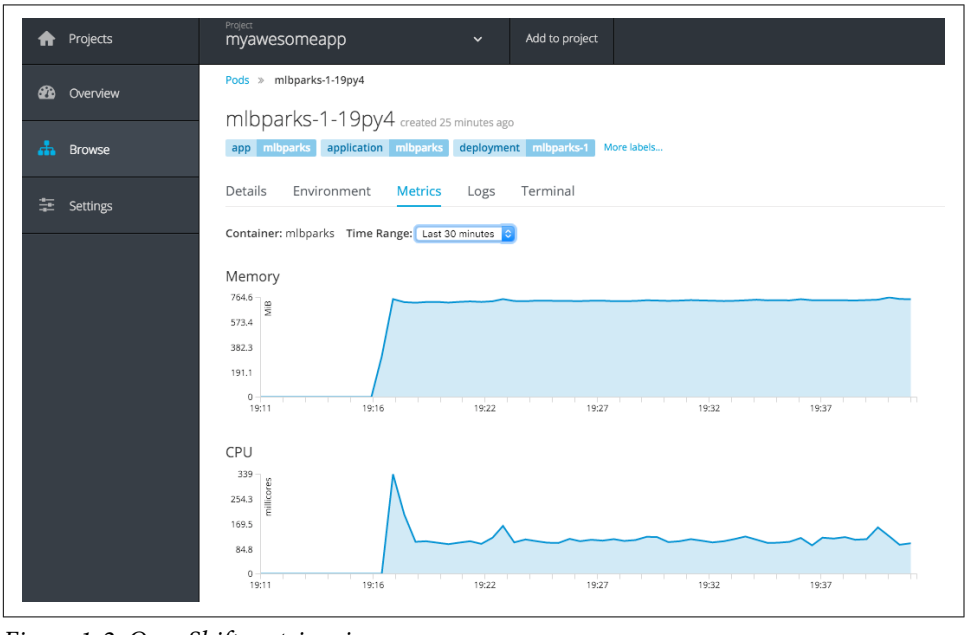

Figure 1-2. OpenShift metrics viewer

Summary

In this chapter we learned about the OpenShift container application platform and the benefits it adds for developers on top of plain old Docker and Kubernetes. There are certainly more advantages to using a full platform than what we have briefly discussed in this chapter. Never fear, as we progress through the contents of this book, we explore more topics in detail to ensure you are successful as a developer using OpenShift.