

# *Pipeline as Code*

CONTINUOUS DELIVERY WITH JENKINS,  
KUBERNETES, AND TERRAFORM

MOHAMED LABOUDRY



MANNING  
SHELTER ISLAND

For online information and ordering of this and other Manning books, please visit [www.manning.com](http://www.manning.com). The publisher offers discounts on this book when ordered in quantity. For more information, please contact

Special Sales Department  
Manning Publications Co.  
20 Baldwin Road  
PO Box 761  
Shelter Island, NY 11964  
Email: [orders@manning.com](mailto:orders@manning.com)

©2021 by Manning Publications Co. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by means electronic, mechanical, photocopying, or otherwise, without prior written permission of the publisher.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in the book, and Manning Publications was aware of a trademark claim, the designations have been printed in initial caps or all caps.

⊗ Recognizing the importance of preserving what has been written, it is Manning's policy to have the books we publish printed on acid-free paper, and we exert our best efforts to that end. Recognizing also our responsibility to conserve the resources of our planet, Manning books are printed on paper that is at least 15 percent recycled and processed without the use of elemental chlorine.

The author and publisher have made every effort to ensure that the information in this book was correct at press time. The author and publisher do not assume and hereby disclaim any liability to any party for any loss, damage, or disruption caused by errors or omissions, whether such errors or omissions result from negligence, accident, or any other cause, or from any usage of the information herein.



Manning Publications Co.  
20 Baldwin Road  
PO Box 761  
Shelter Island, NY 11964

Development editor: Karen Miller  
Technical development editor: Christopher Haupt  
Review editor: Mihaela Batinić  
Production editor: Deirdre S. Hiam  
Copy editor: Sharon Willkey  
Proofreader: Keri Hales  
Technical proofreader: Werner Dijkerman  
Typesetter and cover designer: Marija Tudor

ISBN 9781617297540

Printed in the United States of America

# *brief contents*

---

<b>PART 1</b>	<b>GETTING STARTED WITH JENKINS .....</b>	<b>1</b>
1	■ What's CI/CD? 3	
2	■ Pipeline as code with Jenkins 21	
<b>PART 2</b>	<b>OPERATING A SELF-HEALING JENKINS CLUSTER .....</b>	<b>47</b>
3	■ Defining Jenkins architecture 49	
4	■ Baking machine images with Packer 70	
5	■ Discovering Jenkins as code with Terraform 100	
6	■ Deploying HA Jenkins on multiple cloud providers 140	
<b>PART 3</b>	<b>HANDS-ON CI/CD PIPELINES .....</b>	<b>195</b>
7	■ Defining a pipeline as code for microservices 197	
8	■ Running automated tests with Jenkins 231	
9	■ Building Docker images within a CI pipeline 271	
10	■ Cloud-native applications on Docker Swarm 309	
11	■ Dockerized microservices on K8s 355	
12	■ Lambda-based serverless functions 401	
<b>PART 4</b>	<b>MANAGING, SCALING, AND MONITORING JENKINS .....</b>	<b>439</b>
13	■ Collecting continuous delivery metrics 441	
14	■ Jenkins administration and best practices 467	



# *contents*

---

<i>preface</i>	<i>xiii</i>
<i>acknowledgments</i>	<i>xv</i>
<i>about this book</i>	<i>xvi</i>
<i>about the author</i>	<i>xix</i>
<i>about the cover illustration</i>	<i>xx</i>

## **PART 1     GETTING STARTED WITH JENKINS ..... 1**

### **1    *What's CI/CD?*    3**

1.1	Going cloud native	4
	<i>Monolithic</i>	4
	<i>Microservices</i>	5
	<i>Cloud native</i>	8
	<i>Serverless</i>	10
1.2	Defining continuous integration	12
1.3	Defining continuous deployment	13
1.4	Defining continuous delivery	14
1.5	Embracing CI/CD practices	15
1.6	Using essential CI/CD tools	16
	<i>Choosing a CI/CD tool</i>	17
	<i>Introducing Jenkins</i>	18

### **2    *Pipeline as code with Jenkins*    21**

2.1	Introducing the Jenkinsfile	22
	<i>Blue Ocean plugin</i>	26
	<i>Scripted pipeline</i>	29
	<i>Declarative pipeline</i>	31

2.2	Understanding multibranch pipelines	36
2.3	Exploring the GitFlow branch model	38
2.4	Test-driven development with Jenkins	39
	<i>The Jenkins Replay button</i>	40
	<i>Command-line pipeline linter</i>	41
	<i>IDE integrations</i>	43

## PART 2 OPERATING A SELF-HEALING JENKINS CLUSTER ..... 47

<b>3</b>	<b><i>Defining Jenkins architecture</i></b>	<b>49</b>
3.1	Understanding master-worker architecture	50
3.2	Managing Jenkins workers	52
	<i>SSH</i>	52
	<i>Command line</i>	53
	<i>JNLP</i>	53
	<i>Windows service</i>	54
3.3	Architecting Jenkins for scale in AWS	55
	<i>Preparing the AWS environment</i>	64
	<i>Configuring the AWS CLI</i>	65
	<i>Creating and managing the IAM user</i>	66
<b>4</b>	<b><i>Baking machine images with Packer</i></b>	<b>70</b>
4.1	Immutable infrastructure	71
4.2	Introducing Packer	72
	<i>How does it work?</i>	73
	<i>Installation and configuration</i>	74
	<i>Baking a machine image</i>	75
4.3	Baking the Jenkins master AMI	85
	<i>Configuring Jenkins upon startup</i>	85
	<i>Discovering Jenkins plugins</i>	88
4.4	Baking the Jenkins worker AMI	96
<b>5</b>	<b><i>Discovering Jenkins as code with Terraform</i></b>	<b>100</b>
5.1	Introducing infrastructure as code	101
	<i>Terraform usage</i>	102
5.2	Provisioning an AWS VPC	103
	<i>AWS VPC</i>	104
	<i>VPC subnets</i>	108
	<i>VPC route tables</i>	111
	<i>VPC bastion host</i>	114
5.3	Setting up a self-healing Jenkins master	117
5.4	Running Jenkins with native SSL/HTTPS	124

5.5 Dynamically autoscaling the Jenkins worker pool 128

*Launch configuration* 128 ▪ *Auto Scaling group* 131

*Autoscaling scaling policies* 133 ▪ *Workers CPU*

*utilization load* 136

## 6 *Deploying HA Jenkins on multiple cloud providers* 140

6.1 Google Cloud Platform 141

*Building Jenkins VM images* 141 ▪ *Configuring a GCP network with Terraform* 147 ▪ *Deploying Jenkins on Google Compute Engine* 153 ▪ *Launching automanaged workers on GCP* 157

6.2 Microsoft Azure 162

*Building golden Jenkins VM images in Azure* 162 ▪ *Deploying a private virtual network* 166 ▪ *Deploying a Jenkins master virtual machine* 171 ▪ *Applying autoscaling to Jenkins workers* 178

6.3 DigitalOcean 183

*Creating Jenkins DigitalOcean Snapshots* 183 ▪ *Deploying a Jenkins master Droplet* 186 ▪ *Building Jenkins worker Droplets* 190

## PART 3 HANDBOOK ON CI/CD PIPELINES ..... 195

## 7 *Defining a pipeline as code for microservices* 197

7.1 Introducing microservices-based applications 199

7.2 Defining multibranch pipeline jobs 203

7.3 Git and GitHub integration 205

7.4 Discovering Jenkins jobs' XML configuration 215

7.5 Configuring SSH authentication with Jenkins 219

7.6 Triggering Jenkins builds with GitHub webhooks 222

## 8 *Running automated tests with Jenkins* 231

8.1 Running unit tests inside Docker containers 233

8.2 Automating code linter integration with Jenkins 238

8.3 Generating code coverage reports 240

8.4 Injecting security in the CI pipeline 242

8.5 Running parallel tests with Jenkins 244

8.6 Improving quality with code analysis 246

8.7 Running mocked database tests 248

- 8.8 Generating HTML coverage reports 250
- 8.9 Automating UI testing with Headless Chrome 254
- 8.10 Integrating SonarQube Scanner with Jenkins 260

## 9 *Building Docker images within a CI pipeline 271*

- 9.1 Building Docker images 273
  - Using the Docker DSL* 273 ▪ *Docker build arguments* 277
- 9.2 Deploying a Docker private registry 279
  - Nexus Repository OSS* 279 ▪ *Amazon Elastic Container Registry* 286 ▪ *Azure Container Registry* 288 ▪ *Google Container Registry* 290
- 9.3 Tagging Docker images the right way 291
- 9.4 Scanning Docker images for vulnerabilities 296
- 9.5 Writing a Jenkins declarative pipeline 301
- 9.6 Managing pull requests with Jenkins 305

## 10 *Cloud-native applications on Docker Swarm 309*

- 10.1 Running a distributed Docker Swarm cluster 310
- 10.2 Defining a continuous deployment process 321
- 10.3 Integrating Jenkins with Slack notifications 335
- 10.4 Handling code promotion with Jenkins 341
- 10.5 Implementing the Jenkins delivery pipeline 346

## 11 *Dockerized microservices on K8s 355*

- 11.1 Setting up a Kubernetes cluster 356
- 11.2 Automating continuous deployment flow with Jenkins 360
  - Migrating Docker Compose to K8s manifests with Kompose* 371
- 11.3 Walking through continuous delivery steps 372
- 11.4 Packaging Kubernetes applications with Helm 381
- 11.5 Running post-deployment smoke tests 387
- 11.6 Discovering Jenkins X 390

## 12 *Lambda-based serverless functions 401*

- 12.1 Deploying a Lambda-based application 402
- 12.2 Creating deployment packages 407
  - Mono-repo strategy* 407 ▪ *Multi-repo strategy* 413

12.3	Updating Lambda function code	417
12.4	Hosting a static website on S3	420
12.5	Maintaining multiple Lambda environments	423
12.6	Configuring email notification in Jenkins	434

## PART 4 MANAGING, SCALING, AND MONITORING JENKINS ..... 439

<b>13</b>	<b><i>Collecting continuous delivery metrics</i></b>	<b>441</b>
13.1	Monitoring Jenkins cluster health	442
13.2	Centralized logging for Jenkins logs with ELK	452
	<i>Streaming logs with Filebeat</i>	454
	<i>Streaming logs with the Logstash plugin</i>	461
13.3	Creating alerts based on metrics	462
<b>14</b>	<b><i>Jenkins administration and best practices</i></b>	<b>467</b>
14.1	Exploring Jenkins security and RBAC authorization	468
	<i>Matrix authorization strategy</i>	469
	<i>Role-based authorization strategy</i>	471
14.2	Configuring GitHub OAuth for Jenkins	472
14.3	Keeping track of Jenkins users' actions	475
14.4	Extending Jenkins with shared libraries	476
14.5	Backing up and restoring Jenkins	480
14.6	Setting up cron jobs with Jenkins	484
14.7	Running Jenkins locally as a Docker container	487
	<i>index</i>	493



# ****preface****

---

Ten years ago, I wrote my first makefile to automate the testing, building, and deployment of a C++ application. Three years later, while working as a consultant, I came across Jenkins and Docker and discovered how to take my automation skills to the next level with CI/CD principles.

The beauty of CI/CD is that it's simply a rigorous way of recording what you're already doing. It doesn't fundamentally change how you do something, but it encourages you to record each step in the development process, enabling you and your team to reproduce the entire workflow later at scale. Over the next few months, I started writing blog posts, doing talks, and contributing to CI/CD-related tools.

However, setting up a CI/CD workflow has always been a very manual process for me. It was done via defining a series of individual jobs for the various pipeline tasks through a graphical interface. Each job was configured via web forms—filling in text boxes, selecting entries from drop-down lists, and so forth. And then the series of jobs were strung together, each triggering the next, into a pipeline. This made the troubleshooting experience a nightmare and reverting to the last known configuration in case of failure a tedious operation.

A few years later, the *pipeline-as-code* practice emerged as part of a larger “as code” movement that includes *infrastructure as code*. I could finally configure builds, tests, and deployment in code that is trackable and stored in a centralized Git repository. All the previous pains were alleviated.

I became a fan and believer of pipeline as code, as I transitioned from being a software engineer, tech leader, and senior DevOps manager to now co-leading my first startup as CTO. Pipeline as code became an important part of each project I was part of.

I had the chance to work on different types of architecture—from monolithic, to microservices, to serverless applications—having built and maintained CI/CD pipelines for large-scale applications. Along the way, I accumulated tips and best practices to follow while going through the journey of continuous everything.

The idea of sharing that experience is what triggered this book. Implementing pipeline as code is challenging for many teams, as they require the use of many tools and processes that all work together. The learning curve takes a lot of time and effort, leading people to wonder whether it's worth it. This book is a handbook experience on how to build a CI/CD pipeline from scratch, using the most widely adopted CI solution: Jenkins. I hope the result will help you embrace the new paradigm of building CI/CD pipelines.