Opensource.com

# A guide to Java serverless functions

Why choose Java for serverless application development and how to get started

## ABOUT OPENSOURCE.COM

### What is Opensource.com?

OPENSOURCE.COM publishes stories about creating, adopting, and sharing open source solutions. Visit Opensource.com to learn more about how the open source way is improving technologies, education, business, government, health, law, entertainment, humanitarian efforts, and more.

Submit a story idea: opensource.com/story

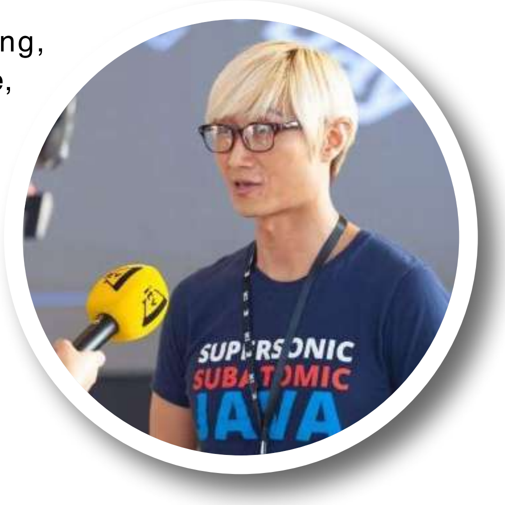
Email us: open@opensource.com

Supported by Red Hat

## DANIEL OH

DANIEL OH Technical Marketing, Developer Advocate, CNCF Ambassador, Public Speaker, Published Author, Quarkus, Red Hat Runtimes.

Follow me at @danieloh30



## INTRODUCTION

## CHAPTERS

# What is serverless with Java?

*Java is still one of the most popular languages for developing enterprise applications. So, why are serverless developers shying away from it?*

FOR DECADES, ENTERPRISES have developed business-critical applications on various platforms, including physical servers, virtual machines, and cloud environments. The one thing these applications have in common across industries is they need to be continuously available (24x7x365) to guarantee stability, reliability, and performance, regardless of demand. Therefore, every enterprise must be responsible for the high costs of maintaining an infrastructure (e.g., CPU, memory, disk, networking, etc.) even if actual resource utilization is less than 50%.

Serverless architecture was developed to help solve these problems. Serverless allows developers to build and run applications on demand, guaranteeing high availability without having to manage servers in multi- and hybrid-cloud environments. Behind the scenes, there are still many servers in the serverless topology, but they are abstracted away from application development. Instead, cloud providers use serverless services for resource management, such as provisioning, maintaining, networking, and scaling server instances.

Because of its effectiveness, the serverless development model is now a requirement for enterprises that want to spin up their business applications on demand rather than running them all the time.

Many open source projects have been created to manage serverless applications on Kubernetes [1] clusters with the Linux container package over virtual machines. The CNCF's Interactive Serverless Landscape [2] is a guide to open source projects, tools, frameworks, and public cloud platforms that enable DevOps teams to handle serverless applications.

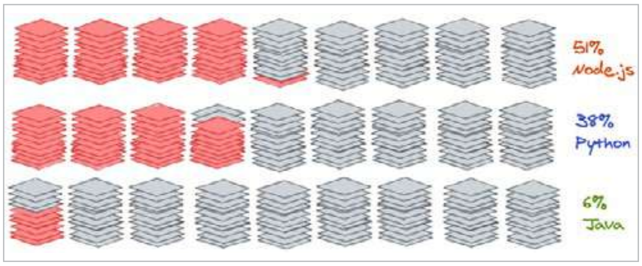
Tools
Security
Framework
Hosted Platform
Installable Platform
dapr

(*CNCF, Apache License 2.0*)

Developers can write code then deploy it quickly to various serverless environments. Then the serverless application responds to demand and automatically scales up and down as needed.

You may be wondering what programming language and runtime are best suited for serverless application development to integrate with the technologies in the figure above. There's not just one answer to this question, but let's take a step back to discuss the application runtime that is most popular for developing business applications in enterprise production environments: Java.

According to Developer Economics [3], as of Q3 2020, more than 8 million enterprise developers are still using Java to achieve their business requirements. Yet, according to a 2020 NewRelic survey [4], Java (at 6%) is clearly not the top choice for forward-thinking developers using a popular cloud service.



*Data from NewRelic's Serverless Benchmark Report (Daniel Oh, CC BY-SA 4.0)*

Resource usage, response times, and latency are critical in serverless development. Serverless offerings from public cloud providers are usually metered on-demand, charged only when a serverless application is up, through an event-driven execution model. Therefore, enterprises don't pay anything when a serverless application is idle or scaled down to zero.

## The state of Java with containers

With this background, you may be asking: "*Why don't developers try to use the Java stack for serverless application development given that existing business applications are most likely developed on Java technologies?*"