Containers All the Way Down!

Optimizing Spring Boot apps for the modern cloud

Mark Heckler

Principal Cloud Advocate, Java/JVM Languages

markheckler@microsoft.com

mark@thehecklers.com

@mkheck

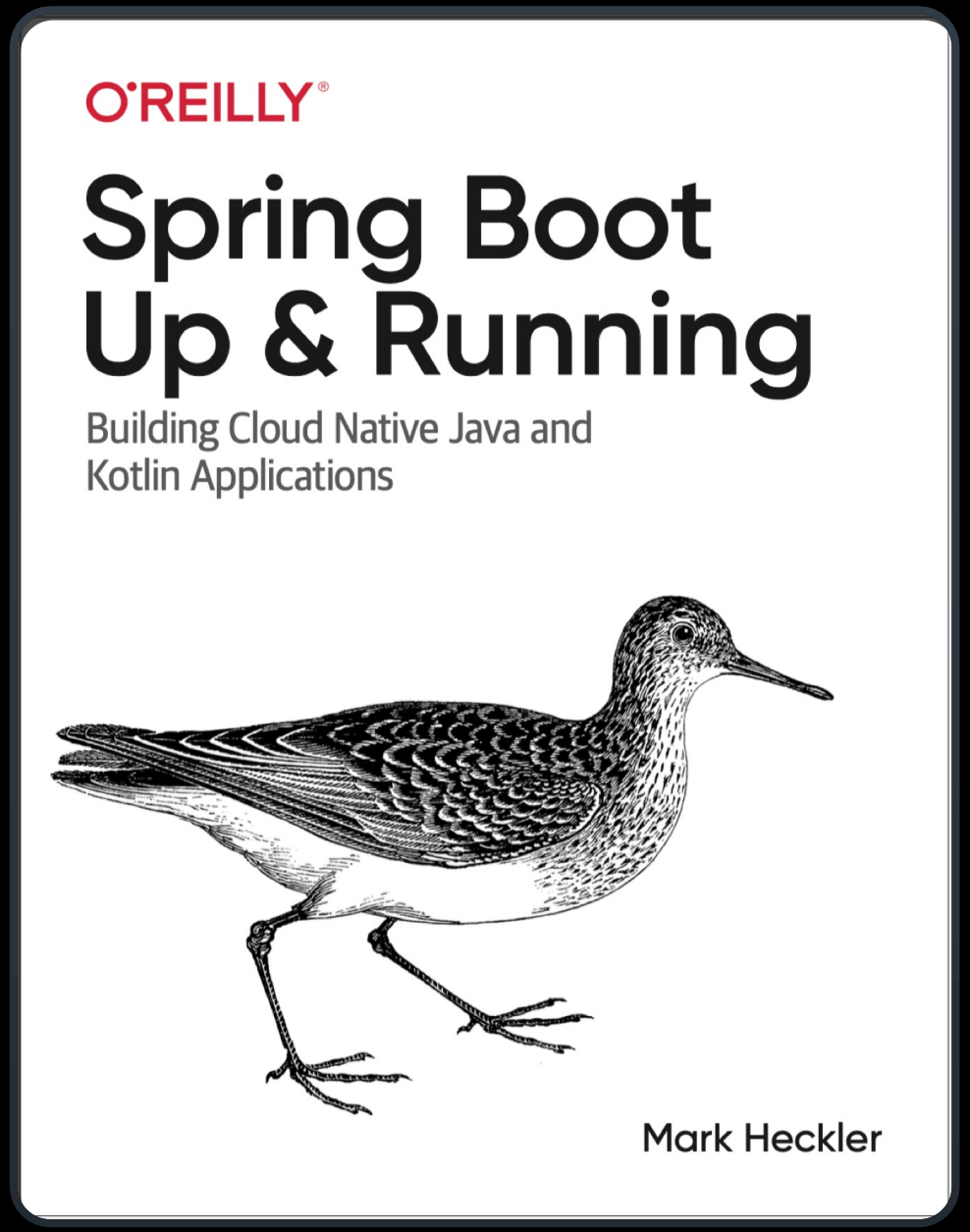
Who am I?

- Architect & Developer
- Principal Advocate
- Author
- Java Champion, Rockstar
- Kotlin Developer Expert
- Pilot 

The book

https://bit.ly/springbootbook

@springbootbook



Why containerize Java apps?

- Portability & consistency in cloud environments
- Solves common deployment challenges
- Enables scalability, security, and cost efficiency

Container images best practices

- Choosing the right base images: OS, distroless, Java
- Optimizing Dockerfiles for Spring Boot
- Multi-stage builds
- Managing environment variables

Spring Boot's built-in containerization support

- Maven: ./mvnw spring-boot:build-image
- Gradle: bootBuildImage
- Cloud Native Buildpacks
- Customizing CNB images

Building for JVM vs. GraalVM native image

- Tradeoffs between JVM and native images
- GraalVM: faster startup, lower resource consumption
- Spring Boot’s GraalVM support for native images

Quick recap

- Containerization is a must for developers
- Containerization is a journey with many potential waypoints/endpoints
- Spring Boot offers great tooling for optimizing your app containers
- A little knowledge can carry you a lot further than most devs
- Learn, apply, profit!

Helpful resources

- https://github.com/mkheck/containersallthewaydown
- https://aka.ms/javalearn (Java containerization refcard)
- markheckler@microsoft.com
- mark@thehecklers.com
- @springbootbook
- @mkheck on X

Thank you for coming!