Booting Up Spring Apps in Lightweight Cloud Services

CON10258

Bruno Borges - @brunoborges
Principal Product Manager
Oracle Cloud PaaS, Oracle Corp.

Josh Long - @starbuxman
Spring Developer Advocate
Spring Team, Pivotal

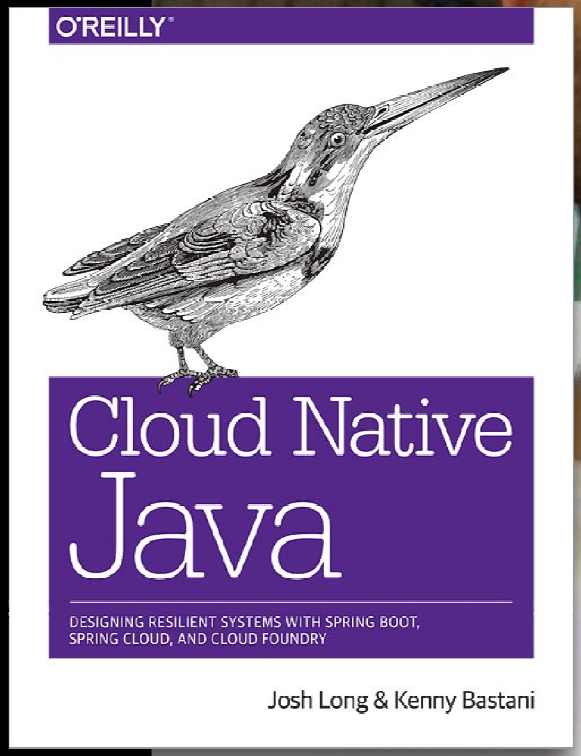
JavaOne™ ORACLE

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Josh Long (龙之春)

the Spring Developer Advocate



- http://**cloudnativejava.io**
- **@starbuxman**
- **jlong**@pivotal.io
- **Java Champion**
- **open-source contributor**
  (Spring Boot, Spring Cloud, Spring Integration, Vaadin, Activiti, etc etc)

Speaker

Bruno Borges

Principal Product Manager
Oracle Cloud Platform

@brunoborges

Program Agenda

1. Lightweight Java Apps in the Cloud
2. (demo) Spring Boot 101
3. (demo) Running Spring Boot Apps on CloudFoundry
4. (demo) Running Spring Boot Apps on Oracle Cloud

Why Move to the Cloud?

Why Move to the Cloud?

Better applications
developed **faster**
and **cheaper**

The Java Ecosystem

Spring Boot

spring

Spring Boot 101, and Demos

Cloud Application Qualities

Stateless / Disposable

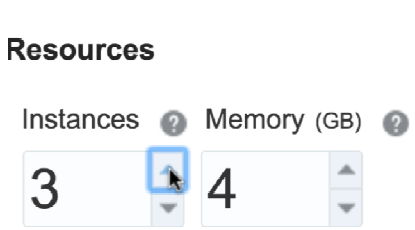
- Applications should be stateless with all persistent data stored in external services like database or key/value stores
- Stateless applications makes scaling easy (esp. when scaling in by disposing of instances)
- Configuration changes will result in the restarting (disposing & creating) of application instances
- Ephemeral disk is useful but there is no guarantee a subsequent request will be handled by the same instance

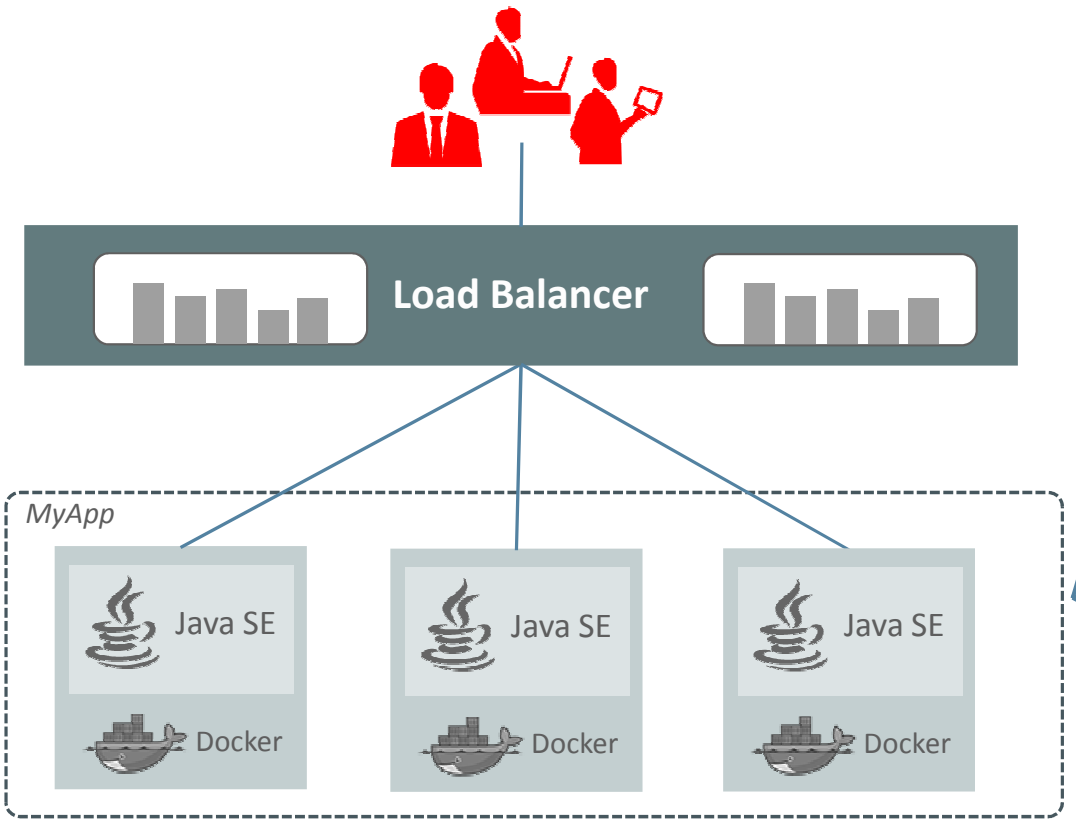
Share Nothing

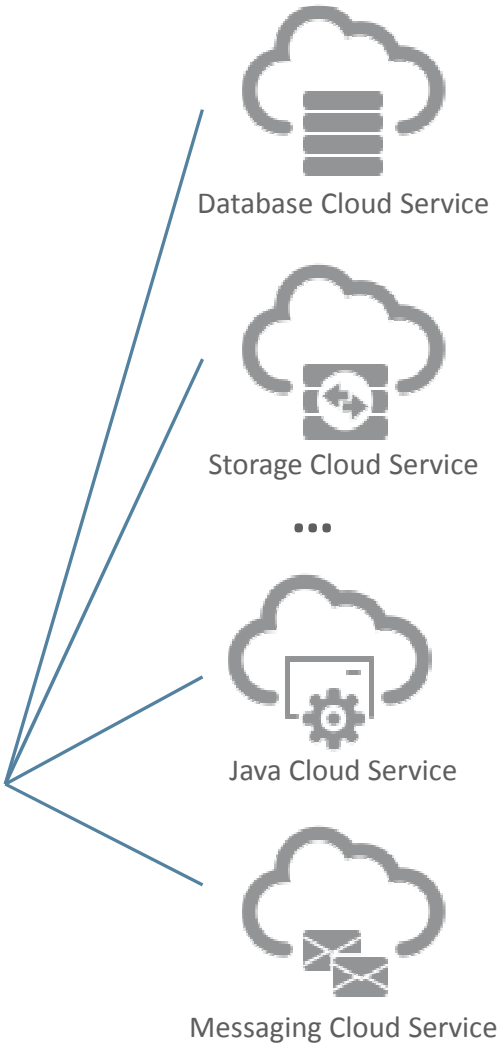
- Ideally, applications should be stateless and share-nothing
- Horizontal scaling of stateless applications by adding instances to handle increased load is simple and reliable
- Not everyone agrees on this point…

Oracle Cloud – App Container Cloud Service

Concurrency in Oracle Java SE Cloud Service







ORACLE

JavaOne ORACLE

Java APaaS Debug Options

Debug locally in emulated environment

- Define environment variables to match target
- Use same JDK release and version as in target
- Use tools like Foreman
- Challenge—”emulated” not identical

Debug in Cloud

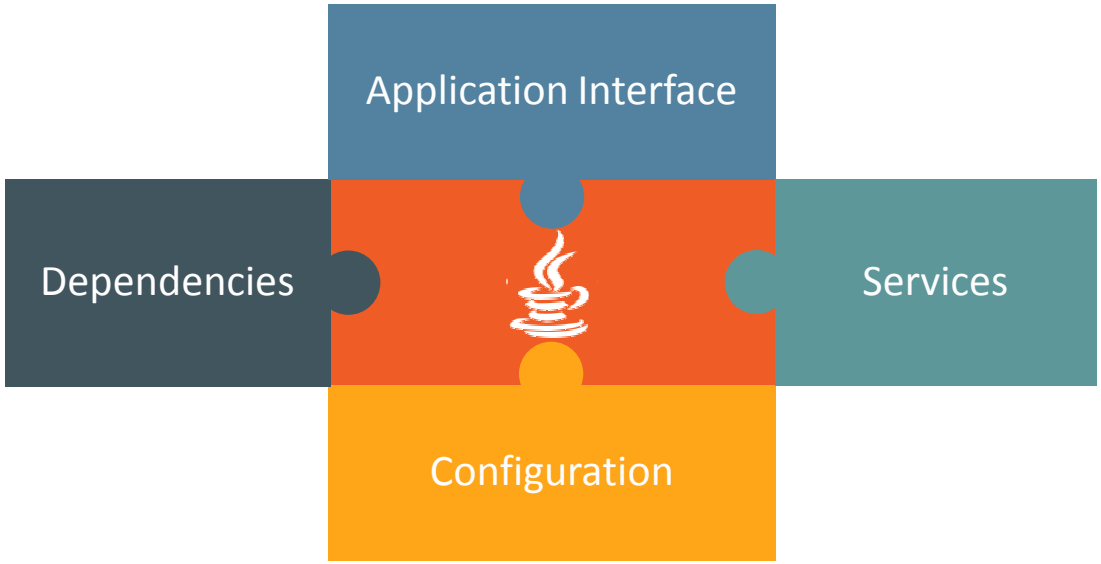
- Restrict to single instance
- Remote test & debug
- E.g., Spring Tools (experimental afaik)
- Challenge—opening up ports and supporting breakpoint callbacks from cloud to desktop

Debug locally using same Docker image as on APaaS

- Same runtime environment as Cloud
- Challenge—dependency on cloud services
- E.g. Heroku (not exactly the same since not Docker in cloud)

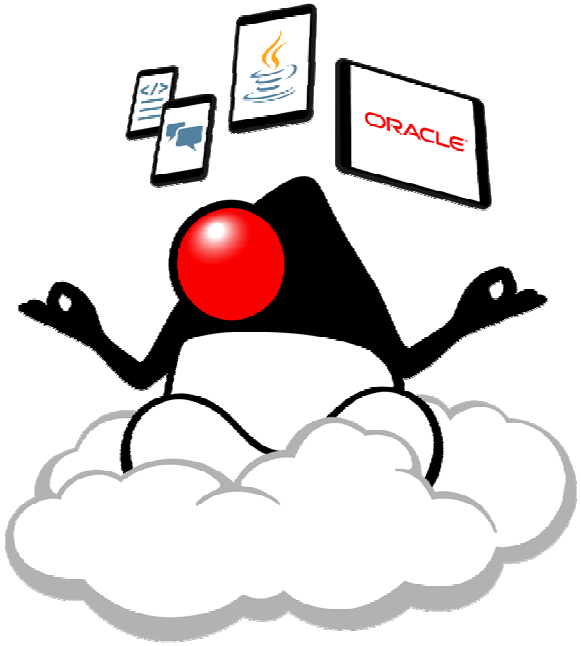
Summary

- Java is one of the best choices for Cloud application development
- Java has broad support from APaaS vendors
- Most of the architectural principals of Cloud apps are simply good practice
  - Configurable
  - Declared Dependencies
  - Pluggable Service
  - …



Java is the Platform!

- Focus on your Java application and not on proprietary platform features
- Architect for platform independence
- Own the stack—don't let vendors dictate
- Use embedded servers, not containers to stay light
- Consider the microservices approach of many small services assembled to provide a complete solution



JavaOne™

ORACLE®