Integration architecture with Java EE and Spring

Markus Eisele (Lightbend)
Josh Long (Pivotal)
@myfear
Josh Long  (龙之春)
the Spring Developer Advocate

- [Cloud Native Java](https://www.amazon.com/Cloud-Native-Java-Resilient-Systems/dp/1491915537)

- [http://cloudnativejava.io](http://cloudnativejava.io)
- [@starbuxman](https://twitter.com/starbuxman)
- josh@joshlong.com
- Java Champion
- open-source contributor
  (Spring Boot, Spring Cloud, Spring Integration, Vaadin, Activiti, etc etc)
Hands-on
Get your environment ready
Where We’ve Been
Motivations for Java EE / J2EE

- Centralized Infrastructures
- Shared baseline
- Centralized governance and management
- Innovation through implementation
- Convention over configuration
Today’s reality for Java EE

- Shared baseline installs no longer relevant
- Customized and distributed fat-jars.
- Innovation can’t be standardized
- Centralized governance vs. DevOps
- Interesting for commodity (e.g. JDBC)
Motivations for Spring

• Spring was born to simplify J2EE APIs
• Spring was born to promote testing, faster feedback loops
• to provide patterns and best practices
• provide flexibility through configuration (over convention)
Today’s reality for Spring

• Java EE (vs J2EE) is very concise, powerful
• Continuous improvement and delivery still key to the power of Spring
• Best practices evolve, and Spring has tried to keep up. Meanwhile, old best practices.. aren’t
• Spring has tried to learn from Java EE and Rails by accommodating smart conventions.
And now?
Java EE \lor \equiv \land \lor \equiv \? \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land \lor \equiv \land
Why this talk?
Use Spring APIs from Java EE
Motivation

• Features that Java EE doesn’t provide out-of-the-box
  • Spring Security
  • Social login
  • the `JdbcTemplate`
  • MVC
  • NoSQL
  • Enterprise Application Integration
  • big data
  • RabbitMQ
  • NoSQL
  • Kafka
  • .... maybe it’s better to show!
## Motivation

[start.spring.io](https://start.spring.io)
Motivation

start.spring.io
Motivation

• Backwards Compatibility
  • Spring framework has a very long tail: Spring framework 4 runs on Servlet 2.5+ (2006!!), Java EE 6 (2009) and Java 6+.
  • Websphere 7 and WebLogic 10.3.4 require JPA 2 feature packs
demo
Coffee Break
Use Java EE APIs from Spring
Motivation

• Want to or have to migrate
• Your team already has the knowledge
• You want to use standards where standards make sense because they’re commoditized or invasive:
  • JTA, JPA, JSR303, JSR 330, JCA, JDBC, JMS, Servlets, etc. etc.
demo
Option 3
Integrate both worlds
Motivation

• Today’s world is growing more and more polyglot and heterogeneous.
• Open Source is driving innovation.
• Closed source stays platform decision.
• Parts move over. Others don’t.
Java EE Application \[ \rightarrow \] REST Calls \[ \rightarrow \] Spring Application
Java EE Application

DB1

DB2

Spring Application
micro services

/ˈmɪkrosvərɪz/ noun
Software Design

Distributed Systems

Outer Architecture

Methodology and Organization

Platform As A Service
Migration Approaches
• evolve or die
• build software that survives, scales and evolves on a dynamic cloud environment

http://cloudnativejava.io/about/
• Understand the challenges of starting a greenfield development vs tearing apart an existing brownfield application into services

• Examine your business domain to see if microservices would be a good fit

• Explore best practices for automation, high availability, data separation, and performance

• Align your development teams around business capabilities and responsibilities

• Inspect design patterns such as aggregator, proxy, pipeline, or shared resources to model service interactions

http://bit.ly/ModernJavaEE
Thank you.

@myfear @starbuxman